TECHNICKÁ UNIVERZITA V LIBERCI
www.tul.cz

Číslicová elektronika

Ondřej Novák
a kolektiv autorů

Liberec 2014

esf evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Bibliografická reference těchto skript:

NOVÁK, O. a kol. *Číslicová elektronika*. 1. vydání. Liberec: Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, 2014. ISBN 978-80-7494-137-5.

DOI: 10.15240/tul/002/2014-11-004

ISBN 978-80-7494-137-5

Tento materiál vznikl v rámci projektu ESF CZ.1.07/2.2.00/28.0050
Modernizace didaktických metod a inovace výuky technických předmětů,
který je spolufinancován Evropským sociálním fondem a státním rozpočtem ČR.

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Obsah

1 Úvod

Tato skripta jsou určena především pro studenty Technické univerzity v Liberci a měla by bých chápána jako jedna z mnoha forem podpory výuky předmětů zaměřených na problematiku číslicové elektroniky. V žádném případě se nejedná o vyčerpávající sbírku informací z této oblasti, nazabýváme se ani fyzikou popisovaných dějů, ani základy analogové elektroniky, která s číslicovou elektronikou velmi úzce souvicí (viz [DOL14]); pochopitelně nejaktuálnější přehled je součástí přednášek příslušných předmětů. Je třeba také říci, že každý z námi zajišťovaných předmětů využívá různě velikou podmnožinu tohoto textu a proto kompletní skripta nejsou v celé své šíři určena všem studentům; osobní zvídavosti však nehodláme nijak bránit... Pokud při svém studiu najdete nějaké nepřesnosti, překlepy atp., kontaktujte libovolného z autorů, abychom mohli zajistit rychlou nápravu.

Úvodní kapitoly skript se zabývají základními pojmy s oblasti číslicové elektroniky, definicí logických stavů a operací, číselnými soustavami atp. Následující kapitola popisují základní kombinační obvody, jejich technologickou realizaci a metodiku návrhu těchto obvodů; dále jsou popsány sekvenční obvody, jejich vlastnosti a metodika jak asynchronních, tak synchronních obvodů. Následující kapitoly pak popisují složitější obvody jako jsou paměti, zakázkové obvody a základní přehled z oblasti mikroprocesorů. Oblast na pomezí číslicové a číslicové elektroniky, tedy převodníky, je popsána v již zmiňovaných skriptech [DOL14].

Závěrem této předmluvy bych rád poděkoval všem, kteří sbírali jednotlivé střípky textu a scelovali je do kompaktních celků kapitol.

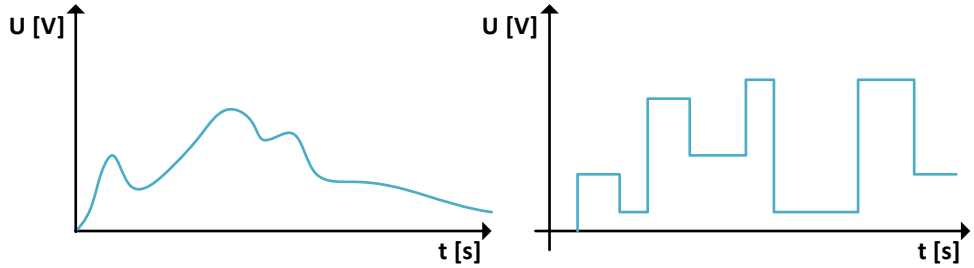
V Liberci 16. 12. 2014
prof. Ing Zdeněk Plíva, Ph.D.

Abecední seznam autorů:
Tomáš Drahoňovský [tomas.drahonovsku@tul.cz]
Ondřej Novák [ondrej.novak@tul.cz]
Jiří Jeníček [jiri.jenicek@tul.cz]
Zbyněk Mader [zbynek.madek@tul.cz]
Petr.Pfeifer [petr.pfeifer@tul.cz]
Zdeněk Plíva [zdenek.pliva@tul.cz]
Martin Rozkovec [martin.rozkovec@tul.cz] – editor skript

2 Základní pojmy

Pro usnadnění komunikace mezi návrháři v oboru číslicové techniky bylo třeba definovat základní pojmy, zavést konvence a normy, které určují jaké fyzikální veličiny a hodnoty jsou přiřazeny logickým veličinám, a jak se s nimi pracuje.

V elektronice používáme dva základní typy signálů. Vedle analogových signálů jsou to signály číslicové (digitální). Analogové signály jsou signály, které se mění spojitě (plynule) v čase. Naproti tomu digitální signály mění svou úroveň nespojitě (skokově). Příklady obou typů signálů můžeme vidět na **Obr. 1**. Číslicové signály jsou vlastně řadou impulsů, která mění nespojitě v čase své úrovně.



Obr. 1: Spojitý (vlevo) a nespojitý signál

2.1 Logické stavy

V číslicové technice pracujeme s fyzikálními veličinami, které je možno při určité míře zjednodušení popsat dvěma stavy. Tyto veličiny pro nás tedy představují proměnné nabývající dvou stavů, hovoříme o binárních proměnných. Příkladem může být obvod tvořený spínačem, žárovkou a napájecím zdrojem (např. **Obr. 3**). Fyzikální veličinou je potom poloha jazýčku spínače, binární proměnnou můžeme nazvat např. SPÍNAČ, stavy této proměnné pak jsou: ZAPNUTO a VYPNUTO. Jiným příkladem je transistor, který v číslicových obvodech přechází mezi dvěma stavy: UZAVŘENÍ a SATURACE.

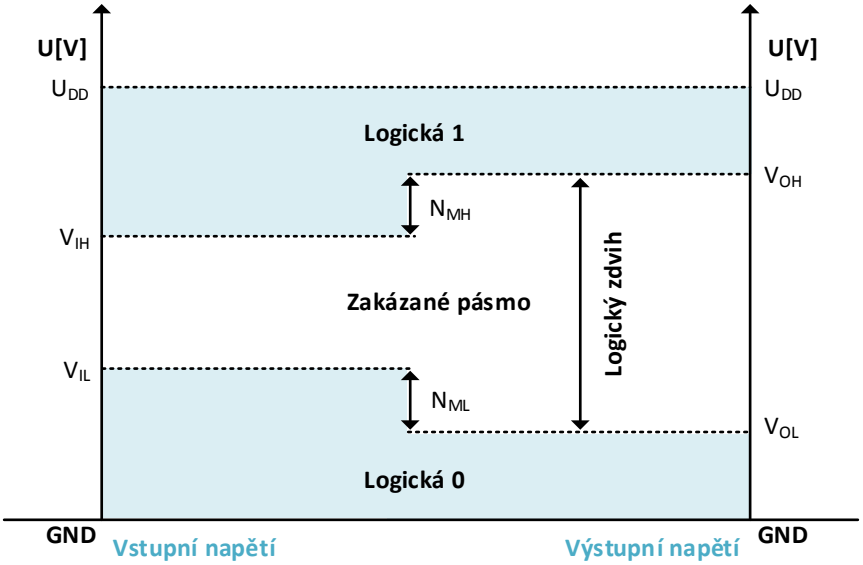
Logickou konvencí je v uvedeném příkladu spínače to, že stavy ZAPNUTO a VYPNUTO nahradíme logickými hodnotami ‘nepravda’ a ‘pravda‘, symboly '0' a '1' nebo anglickými výrazy 'false' a 'true'. Někdy pro zdůraznění faktu, že pracujeme s logickými veličinami, píšeme místo '0' a '1' log. 0 a log. 1. Přiřazení mezi stavy a logickými hodnotami je libovolné.

Stavu ZAPNUTO může odpovídat log 0 i log 1. Častěji se však volí přiřazení: ZAPNUTO = '1' a VYPNUTO = '0'. V případě, že logickou hodnotu přiřazujeme k úrovni napětí, potom v pozitivní logice volíme přiřazení: nízké napětí = '0' a vyšší napětí = '1'. V negativní logice přiřazujeme napětí k logickým hodnotám opačně. V anglické terminologii označujeme nízkou úroveň napětí zkratkou L (Low) a vysokou úroveň napětí zkratkou H (High).

V technické praxi je třeba bezpečně rozlišit jednotlivé binární stavy. Typické přiřazení napěťových úrovní je znázorněno na Obr. 2. Na tomto obrázku vidíme, že mezi typickými napěťovými úrovněmi pro log. 0 a log. 1 je pásmo zakázaných napěťových úrovní. Toto pásmo se nazývá Logický zdvih a šířka tohoto pásma určuje míru bezpečnosti rozpoznání logických úrovní.

2.2 Logické operace

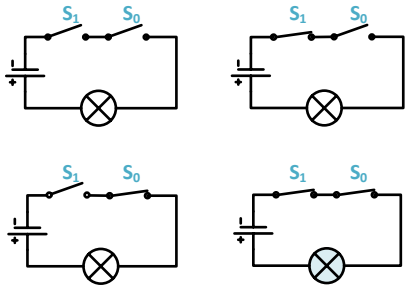
Pro binární proměnné je možno definovat logické operace. Tyto operace se zpravidla popisují pomocí logických operátorů (log. součet, log. součin, negace, …) nebo je můžeme popsat tzv. pravdivostní tabulkou.



Obr. 2: Přiřazení napěťových úrovní binárním hodnotám pro standard TTL

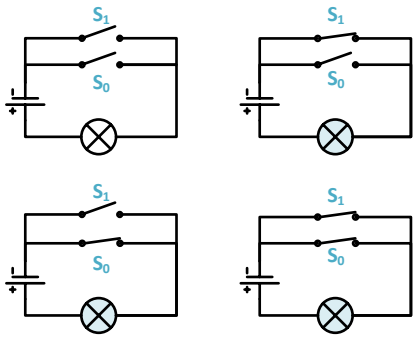
Obr. 3 a **Obr. 4** ukazují obvod se dvěma spínači zařazenými a odpovídající pravdivostní tabulku. V tomto příkladu jsou dvě proměnné vstupní (S_0 a S_1) a jedna proměnná výstupní (Žárovka).

Pravdivostní tabulka udává vztah mezi vstupními a výstupními proměnnými, tj. je předpisem, který určuje, jak je možno výstupní proměnnou řídit pomocí vstupů. Logické operace mezi proměnnými S_0 a S_1 jsou logický součin (**Obr. 3**) a logický součet (**Obr. 4**). Obvod spínačů a žárovky potom modeluje logickou operaci součinu (součtu) a my o něm hovoříme jako o logickém členu.



S_0	S_1	Žárovka
0	0	0
0	1	0
1	0	0
1	1	1

Obr. 3: Obvodové schéma realizující funkci logického součinu (AND) a pravdivostní tabulka



S_0	S_1	Žárovka
0	0	0
0	1	1
1	0	1
1	1	1

Obr. 4: Obvodové schéma realizující funkci logického součtu (OR) a pravdivostní tabulka

Logické členy tvoří logický obvod.

2.3 Číselné soustavy

S číslicovými signály je třeba provádět nejenom logické ale i aritmetické operace. Abychom porozuměli těmto operacím, uvedeme zde stručné vysvětlení způsobu převodů mezi desítkovou, binární a hexadecimální soustavou, které se v číslicových obvodech používají nejčastěji. Libovolné kladné číslo F(Z) lze zapsat pomocí mnohočlenu ve tvaru:

$$F(Z) = \sum_{i=0}^{m-1} a_i Z^i$$

Kde F(Z) je číslo vyjádřené v číselné soustavě o základu Z, ai jsou číselné koeficienty, m je počet řádových míst. Pro jednoduchost v číselných soustavách zapisujeme pouze koeficienty ai. V praxi potom často řešíme problém, který koeficient vyjadřuje nejvyšší řád a který nejnižší. Zpravidla se řídíme konvencí, která říká, že nejvyšší řád je vlevo a nejnižší vpravo. U dvojkové soustavy nejvyšší řád označujeme jako MSB (Most Significant Bit) a nejnižší řád jako LSB (Least Significant Bit).

Tab. 1: Vyjádření čísel od 0 do 15 v různých soustavách

Číslo (dekadicky)	Základ 2	Základ 8	Základ 16
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Abychom rozlišili jednotlivé soustavy mezi sebou, používáme v případech, kdy by mohlo dojít k nejasnosti, o kterou soustavu se jedná, označení (2) nebo b pro dvojkovou (binární) soustavu, (8) nebo o pro osmičkovou (oktalovou) soustavu, (10) nebo d pro desítkovou (dekadickou) soustavu a (16) nebo h pro šestnáctkovou (hexadecimální) soustavu. Příklady čísel: 011(2), 011b, 47(8), 0A23(16), 0A123h nebo 011(2), 011b, 47(8), 0A23(16), 0A123h.

Vyjádřete číslo 275(10) binárně.

Číslo v desítkové soustavě 275(10) je možno vyjádřit jako $2\times10^3 + 7\times10^1 + 5\times10$. Číslo 1101(2) ve dvojkové soustavě vyjadřuje hodnotu $1\times2^3 + 1\times2 + 0\times2^1 + 1\times2^0$, neboť předpokládáme, že MSB je vlevo.

2.4 Záporná čísla

V případě, kdy chceme rozlišit kladná a záporná čísla ve dvojkové soustavě použijeme znaménko mínus stejně jako v soustavě dekadické. Pro kódování znaménka v počítači se používají různé způsoby. Nejčastěji jsou to tyto tři:

- Vyhrazení znaménkového bitu
- Přičtení konstanty
- S využitím dvojkového doplňku

V případě kdy jeden bit (většinou ten nejvýznamnější) představuje znaménko, ostatní bity představují absolutní hodnotu daného čísla. Konvence říká, že v případě, že znaménkový bit je roven jedné, jedná se o záporné číslo a je-li roven nule, jedná se o číslo kladné. Znaménkový bit nám snižuje rozsah čísel, která můžeme zakódovat daným počtem bitů. Např. pomocí 8mi bitů můžeme vyjádřit čísla v rozsahu -127 až 127 (přičemž máme kladnou a zápornou nulu).

Přičtení konstanty

Při využití tohoto způsobu kódování záporných binárních čísel dochází k posouvání nuly tím, že ke každému číslu přičteme vhodnou konstantu. Např. pro čísla v rozsahu -127 až 128 bychom volili konstantu 127.

Dvojkový doplněk

Dvojkový doplněk binárního čísla získáme jako jeho inverzi zvětšenou o jedničku. Kódování záporných čísel pomocí dvojkového doplňku nám umožňuje snadno provádět odčítání ve dvojkové soustavě (přičtením záporného čísla).

Rozsah čísel, který je možné pomocí dvojkového doplňku zakódovat je (-2^{n-1}) až $(2^{n-1} -1)$, to znamená, že pomocí např. 4 bitů můžeme zakódovat čísla v rozsahu: -8 až 7, pomocí 8mi bitů čísla v rozsahu: -128 až 127 atd.

Při převodu kladného čísla do dvojkového doplňku musíme nejprve kladné binární číslo doplnit zleva nulami na správný počet bitů (např. číslo $5_{10} = 101_2$ doplníme na 0101_2). V případě že tak neučiníme, nedostaneme správnou hodnotu.

Tab. 2: Vyjádření čísel ve dvojkovém doplňku

Dvojkový doplněk	Dekadické číslo
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6

Dvojkový doplněk	Dekadické číslo
1001	-7
1000	-8

Vypočtěte dvojkový doplněk k číslu $1110_2 = 14_{10}$.

Číslo 1110_2 nejprve doplníme na potřebný počet bitů (pro vyjádření čísla -14 potřebujeme 5 bitů) a tím dostaneme 01110. Inverze tohoto čísla je 10001. Přičtením jednotky v LSB dostaneme číslo $10010_2 = -14_{10}$. Uvedené číslo představuje záporné číslo s absolutní hodnotou shodnou s číslem původním.

2.5 Převody mezi číselnými soustavami

Přepočet čísla z libovolné soustavy do soustavy se základem 10 provedeme dosazením do mnohočlenu, uvedeného výše.

Převeďme číslo 11001(2) do dekadické soustavy

$11001(2) = 1\times2^4 + 1\times2^3 + 0\times2^2 + 0\times2^1 + 1\times2^0 = 16 + 8 + 1 = 25(10).$

Přepočet z desítkové soustavy do ostatních soustav se provádí pomocí algoritmu postupného dělení čísla základem nové soustavy.

Převeďme číslo 25(10) do dvojkové soustavy.

Řešení:

25 : 2 = 12, zbytek 1 .... $a_0 = 1$

12 : 2 = 6, zbytek 0 .... $a_1 = 0$

6 : 2 = 3, zbytek 0 .... $a_2 = 0$

3 : 2 = 1, zbytek 1 .... $a_3 = 1$

1 : 2 = 0, zbytek 1 .... $a_4 = 1$

Při dělení získáváme zbytky, které reprezentují výsledné bity binárního čísla postupně od LSB. Postupné dělení provádíme tak dlouho, až dostaneme nulový výsledek částečného dělení. Výsledek: 25(10) = 11001(2).

Přepočet mezi dvojkovou, osmičkovou a šestnáctkovou soustavou. Využíváme toho, že 8 i 16 jsou mocninou čísla 2. Z toho vyplývá, že jeden řád osmičkové soustavy je representován třemi místy dvojkovými, jedno místo šestnáctkové čtyřmi místy dvojkovými a opačně.

Převeďte číslo 1010111111010001(2) do šestnáctkové a osmičkové soustavy.

Pro převod do šestnáctkové soustavy rozdělíme číslo na čtveřice od nejnižšího řádu a čtveřice nahradíme šestnáctkovou cifrou.

0001	0101	1111	1101	0001
1	5	F	D	1

Výsledek: 1010111111010001(2) = 15FD1(16)

Pro převod do osmičkové soustavy rozdělíme číslo na trojice od nejnižšího řádu a ty pak nahradíme osmičkovou cifrou

010	101	111	111	010	001
2	5	7	7	2	1

Výsledek:1010111111010001(2) = 257721(8).

2.6 Sčítání a odčítání ve dvojkové soustavě

a) Sčítání: Při sčítání se příslušné koeficienty ai sčítají obdobně jako v desítkové soustavě:

00011011 (2)

00110010 (2)

01001101 (2)

b) Odčítání: Ručně provádíme odčítání v číselných soustavách tak, jak jsme zvyklí ze soustavy desítkové, respektujeme při tom změny řádu v dané soustavě.

01001011 (2)

-00110010 (2)

00011001 (2)

Výpočetní technika používá pro odčítání převodu na přičítání dvojkového doplňku menšitele. Dvojkový doplněk k menšiteli 00110010(2) získáme jako jeho inverzi (11001101) zvětšenou o 1 (11001110). Dvojkový doplněk reprezentuje záporné číslo o stejné absolutní hodnotě, jako bylo číslo původní. Dvojkový doplněk přičteme k menšenci:

01001011 (2)

11001110 (2)

00011001 (2)

Nula v řádu MSB signalizuje, že výsledek odečítání je kladné číslo. V případě, že výsledek odečítání by byl záporný, získali bychom výsledek s MSB rovným jedné. V případě, že bychom chtěli získat absolutní hodnotu tohoto záporného čísla, určili bychom ji jako dvojkový doplněk výsledku.

Ve dvojkové soustavě proveďte operaci odečtení čísel 1-7.

Pro binární kódování čísel v rozsahu od –7 do +7 využijeme 4 bitů. Při využití dvojkového doplňku čísla 7 provedeme následující výpočet:

0001 (2)

1001 (2)

1010 (2)

Výsledek 1010 je vyjádřením čísla –6. Absolutní hodnotu získáme jako jeho inverzi (0101) a přičtení jednotky (0110).

2.7 Další používané kódy

Doposud jsme hovořili o binární, oktalové, dekadické a hexadecimální číselné soustavě.

Tyto soustavy spolu s pravidly, která říkají, jaké informace jsou přiřazeny jednotlivým číslům, nazýváme kódy. Příkladem kódu může být tzv. doplňkový kód, který jsme využili při odečítání binárních čísel.

Pro snazší práci s čísly desítkové soustavy byl vytvořen BCD (Binary Coded Decimal) kód. Tento kód slouží k zobrazení tzv. desítkových číslic, tj. čísel 0, 1, … , 9. Tento kód je 4 bitový, neboť k zakódování každé z deseti číslic vyžaduje 4 bity. V Tab. 3 je tento kód uveden.

Tab. 3: Tabulka BCD kódu

Znak	Obraz			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1

V dalším textu popíšeme ještě kód Grayův. Tento kód má tu vlastnost, že sousední čísla se liší pouze v jednom bitu. Této výhodné vlastnosti je využito např. při sestavování Karnaughovy mapy v kapitole (kapitola K. mapy). Tabulka Grayova kódu pro čísla od 0 do 15 je uvedena v Tab. 4.

Tab. 4: Grayův kód

Znak	Obraz			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1
10	1	1	1	1
11	1	1	1	0
12	1	0	1	0
13	1	0	1	1
14	1	0	0	1
15	1	0	0	0

Kód 1 z N má tu vlastnost, že obraz je tvořen N-1 nulami a jednou jedničkou, která je na pozici dané vstupním znakem. Například číslo 3 se zobrazí v kódu 1 z 8 jako vektor 00010000. Kód se využívá například u dekodéru popsaného v kap. 4.2.

Tab. 5: Kód 1 z N

Znak	Obraz			
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0

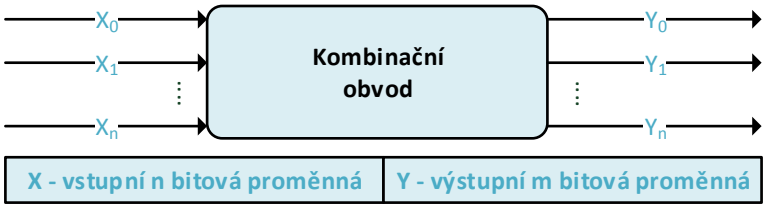
Důležitou skupinou kódů jsou alfanumerické kódy. Tyto kódy zobrazují abecedně číslicovou informaci pomocí binárních čísel. Alfanumerické kódy se začaly používat pro přenos zpráv a byly pro tento účel normalizovány. Nejčastěji se používají kódy osmibitové, tzn. každý znak, který se přenáší, je kódován osmi bity. Velmi rozšířený je kód ASCII, který je uzpůsoben k přenosu anglické abecedy a dalších běžných znaků je nevhodný k přenosu znaků české abecedy. Z tohoto důvodu byl upraven pro přenos těchto českých znaků, čímž vznikl kód Kamenických, později LATIN 2.

3 LOGICKÉ OBVODY

Logické obvody mohou být založeny na různém fyzikálním principu. Mohou být mechanické, pneumatické, elektrické, elektronické, magnetické, optické, atd. V tomto textu se zabýváme, elektronickými logickými obvody, jedná se totiž o nejčastěji používaný fyzikální princip. V případě, že chceme měřit jinou fyzikální veličinu (např. teplotu, tlak, polohu, ...) popřípadě chceme vytvořit neelektronickou akční veličinu (síla, poloha, teplota, osvětlení, ...) používáme snímače a převodníky, které umožní zpracování informací elektronickými logickými obvody. Z hlediska chování logické obvody rozdělujeme na kombinační a sekvenční. Sekvenční obvody pak mohou být dále rozděleny na asynchronní a synchronní. V následujícím textu se budeme nejprve zabývat kombinačními obvody.

3.1 KOMBINAČNÍ LOGICKÉ OBVODY

Ideální kombinační obvod nemá žádnou vnitřní paměť. Odezva ideálního kombinačního obvodu v určitém časovém okamžiku je podmíněna pouze těmi hodnotami, které jsou v uvažovaném okamžiku na vstupech obvodu, a je okamžitá. U reálného kombinačního obvodu je třeba uvažovat zpoždění odezvy na změnu vstupů. Při dodržení návrhových pravidel pro konstrukci obvodů však můžeme toto zpoždění zanedbat bez toho, aby došlo ke změně předpokládané funkce.



Obr. 5: Kombinační obvod - výstupní proměnná je jednoznačně určena proměnnou vstupní

Pomocí kombinačních obvodů můžeme realizovat logické a aritmetické funkce, jejichž argumenty jsou vstupní proměnné a výsledky operací jsou dány proměnnými výstupními.

3.1.1 ZÁKLADNÍ LOGICKÉ FUNKCE

Příklad obvodu, který realizuje funkci součinu a příslušná pravdivostní tabulka je na Obr. 3. Funkce může mít i větší počet operandů než dva. Funkční hodnota je rovna '1' pouze v případě, že na všechny vstupy přivedeme hodnotu '1'. Algebraicky je funkce vyjádřena pomocí symbolu tečka: (Y = A·B). Anglická zkratka funkce je AND.

Příklad obvodu, který realizuje funkci součtu a příslušná pravdivostní tabulka je na Obr. 4. Funkce může mít libovolný počet operandů ne menší než dva. Funkční hodnota je rovna '0' pouze v případě, že na všechny vstupy přivedeme hodnotu: '0'. Algebraicky vyjadřujeme funkci pomocí znaménka plus: (Y = A+B). Anglická zkratka funkce je OR.

Negace je unární operace, to znamená, že má pouze jeden operand. Funkce negace mění hodnotu proměnné z '0' na '1' a opačně. Algebraicky vyjadřujeme funkci pomocí svislé čáry nad logickou proměnnou popř. symbolem apostrof (´,): (Y = $\overline{A}$, Y = ‘A, Y = A´) Anglická zkratka funkce je INVERT, NON, nebo NOT. V grafické reprezentaci funkcí značíme negaci symbolem kroužku. Symbol kroužku může být v libovolné části schématu obvodu. Jestliže je umístěn na vstupu schématické značky, interpretujeme jej jako negaci příslušné vstupní proměnné, pokud je na výstupu jedná se o negaci výstupní proměnné.

Funkce může mít libovolný počet operandů ne menší než dva. Funkční hodnota je rovna '0'

pouze v případě, že na všechny vstupy přivedeme hodnotu '1'. Algebraicky vyjadřujeme funkci pomocí tečky (popřípadě tečku vynecháme) a pomocí pruhu popř. (‘): (Y = $\overline{A \cdot B}$, Y = $\overline{AB}$, Y = ‘(A·B)). Anglická zkratka funkce je NAND.

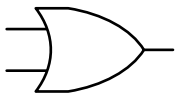
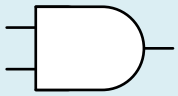
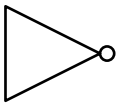
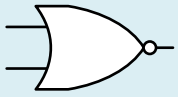
Funkce může mít libovolný počet operandů ne menší než dva. Funkční hodnota je rovna '1' pouze v případě, že na všechny vstupy přivedeme hodnotu '0'. Algebraicky vyjadřujeme funkci pomocí znaménka plus a pomocí pruhu popř. (‘): (Y = $\overline{A+B}$, Y = ‘(A+B)). Anglická zkratka funkce je NOR.

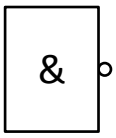
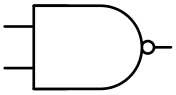
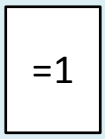
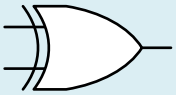
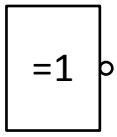
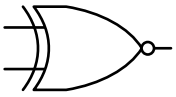
Funkce má dvě proměnné. Funkční hodnota je rovna '1' pouze v případě, že právě na jeden vstup přivedeme '1'. Algebraicky vyjadřujeme funkci pomocí znaménka ⊕: (Y = A⊕B). Anglická zkratka funkce je XOR.

Funkce má dvě proměnné. Funkční hodnota je rovna '0' pouze v případě, že právě na jeden vstup přivedeme '1'. Algebraicky vyjadřujeme funkci pomocí znaménka⊕ a pruhu: (Y = $\overline{A \oplus B}$). Anglická zkratka funkce je XNOR (EXCLUSIVE NOR).

Uvedené funkce je možno znázornit nejen algebraicky a slovně ale také i graficky. Tyto symboly jsou uvedeny v Tab. 5. Každý symbol podle ČSN má nalevo vstupy a napravo výstupy, uvnitř obdélníčků je vepsán symbol funkce. Pospojováním symbolů čarami je možno přehledně znázornit složitější funkce. V současnosti jsou u nás využívány i symboly odpovídající americkým standardům, které umožňují symbol libovolně natáčet, což může vést u složitějších schémat k větší přehlednosti.

Tab. 6: Přiřazení grafických symbolů podle ČSN a podle anglo-americké konvence jednotlivým logickým členům a jejich pravdivostní tabulky

Funkce	Anglický název	ČSN	Anglo-Americká konvence	Pravdivostní tabulka
Logický součet	OR	≥1		a b y 0 0 0 0 1 1 1 0 1 1 1 1
Logický součin	AND	&		a b y 0 0 0 0 1 0 1 0 0 1 1 1
Negace	NOT	1		b y 0 1 1 0
Negovaný součet	NOR	≥1		a b y 0 0 1 0 1 0 1 0 0 1 1 0

Funkce	Anglický název	ČSN	Anglo-Americká konvence	Pravdivostní tabulka
Negovaný součin	NAND			a b y 0 0 1 0 1 1 1 0 1 1 1 0
Nonekvivalence	XOR			a b y 0 0 0 0 1 1 1 0 1 1 1 0
Ekvivalence	XNOR			a b y 0 0 1 0 1 0 1 0 0 1 1 1

Ve výše uvedených odstavcích jsme popsali nejužívanější logické funkce. Seznam však není úplný, např. pro dvě proměnné existuje 16 možných různých funkcí. Libovolnou funkci však můžeme vytvořit sestavením z několika základních funkcí. Takováto skupina funkcí se nazývá úplným souborem logických funkcí. Tvoří ji například dvojice funkcí: AND + NON, OR + NOT nebo dokonce může být tvořena samostatnou funkcí jako například funkcí NAND nebo NOR. Tohoto poznatku se využívá u stavebnic integrovaných obvodů, kde je možno skládat jednotlivé integrované obvody obsahující logické obvody pouze s funkcí NAND tak, že pomocí nich realizujeme libovolné složitější zapojení. Symboly složitějších logických funkcí se vytvářejí podobně jako symboly funkcí základních. Symbolická značka je zpravidla svislými pruhy rozdělena na tři pole: pole vstupů, pole vlastní funkce a pole výstupů. Vstupní a výstupní pole je ještě rozděleno vodorovnými čarami na skupiny jednotlivých proměnných, které mají podobnou funkci (viz např. Obr. 25).

Booleova algebra

Z matematiky víme, že logické výrazy je možno upravovat pomocí zákonů Booleovy algebry. Tyto zákony je dobré znát, protože nám umožní převádět logické výrazy na takové, které mají požadované vlastnosti (například minimální počet součtových členů), realizovatelnost pomocí jednoho typu hradel atd. Připomeňme ještě, že v Booleově algebře jsou dvě základní binární operace - logický součet (+) a součin (·), unární operace negace (značeno pruhem) a dvě nulární operace (konstanty) - logická 0 a 1. Základní axiomy jsou uvedeny v Tab. 6, odvozené výrazy jsou uvedeny v Tab. 7. Uvedené zákony se s výhodou využívají pro optimalizaci logických výrazů, De Morganovy zákony umožňují převádět logické výrazy ze součinového tvaru na součtový a opačně (tj. např. převod z logického obvodu obsahujícího pouze hradla NOR na obvod obsahující pouze hradla NAND).

Tab. 7: Booleova algebra - Axiomy

Základní axiom	Součet	Součin
Komutativita	$a+b=b+a$	$a \cdot b=b \cdot a$

Základní axiom	Součet	Součin
Asociativita	$a+(b+c)=(a+b)+c$	$a(b\cdot c)=(a\cdot b)c$
Distributivita	$a+(b\cdot c)=(a+b)(a+c)$	$a(b+c)=(a\cdot b)+(a\cdot c)$
Neutralita 0 a 1	$a+0=a$	$a\cdot 1=a$
Vlastnosti komplementu	$a+\overline{a}=1$	$a\cdot\overline{a}=0$
Agresivita 0 a 1	$a\cdot 0=0$	$a+1=1$
Idempotence	$a\cdot a=a$	$a+\overline{a}=1$
Absorbce	$a+a\cdot b=a$	$a(a+b)=a$

Z těchto základních zákonů byly odvozeny následující zákony:

Tab. 8: Booleova algebra - odvozené zákony

Pravidlo	Příklad	
Dvojí negace	$\overline{\overline{a}}=a$	
Absorpce negace	$a+\overline{a}\cdot b=a+b$	$a(\overline{a}+b)=a\cdot b$
De Morgan	$\overline{(a+b)}=\overline{a}\cdot\overline{b}$	$\overline{a\cdot b}=\overline{a}+\overline{b}$
Consensus	$a\cdot b+\overline{a}c+b\cdot c=ab+\overline{a}c$	$(a+b)(a+c)(b+c)=(a+b)+(\overline{a}+c)$

3.1.2 Pravdivostní tabulka

Logické funkce je možné vyjádřit kromě algebraického popisu také pravdivostní tabulkou. V této kapitole si vlastnosti pravdivostní tabulky rozebereme podrobněji.

Tab. 9: Pravdivostní tabulka funkce f. Vstupní proměnné a, b, c. Bit c je MSB bitem.

Stavový Index S	c	b	a	Funkční hodnota F(c,b,a)	Minterm PS
0	0	0	0	0	$\overline{c}\cdot\overline{b}\cdot\overline{a}$
1	0	0	1	1	$\overline{c}\cdot\overline{b}\cdot a$
2	0	1	0	1	$\overline{c}\cdot b\cdot\overline{a}$
3	0	1	1	0	$\overline{c}\cdot b\cdot a$
4	1	0	0	1	$c\cdot\overline{b}\cdot\overline{a}$
5	1	0	1	0	$c\cdot\overline{b}\cdot a$
6	1	1	0	1	$c\cdot b\cdot\overline{a}$
7	1	1	1	0	$c\cdot b\cdot a$

V Tab. 8 je označen první sloupec jak stavový index. Ukazuje současně číslo řádku tabulky a binární hodnotu čísla tvořeného vstupními proměnnými, jestliže jsou seřazeny vzestupně (tj. bity s nejvyšší vahou nalevo). Druhý sloupec obsahuje všechny možné kombinace vstupních hodnot jednotlivých proměnných, třetí sloupec funkční hodnoty příslušné odpovídajícím hodnotám vstupů. Sloupec minterm nám ukazuje, jak je možno jednotlivým řádkům tabulky přiřadit logický výraz. Například jestliže tabulka definuje funkci nabývající hodnoty 1 v řádcích 1, 2, 4 a 6, potom z posledního sloupce můžeme odvodit algebraické vyjádření funkce f:

$$f=\overline{c}\cdot\overline{b}\cdot a+\overline{c}\cdot b\cdot\overline{a}+c\cdot\overline{b}\cdot\overline{a}+c\cdot b\cdot\overline{a}$$

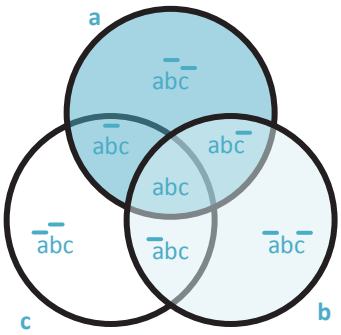
V pravdivostní tabulce se může vyskytovat symbol X a to jak ve sloupci vstupů tak funkčních hodnot. Jeho význam je „nedefinovaná hodnota“. Používáme jej tehdy, když na uvedeném vstupu resp. výstupu nezáleží. Zápis pravdivostní tabulky s využitím X šetří místo. V Tab. 9 je stejná funkce jako v předchozí pravdivostní tabulce, vidíme však, že zápis je úspornější.

Tab. 10: Pravdivostní tabulka funkce f, využití nedefinovaných hodnto X.

STAV. INDEX	C B A	F
0	0 0 0	0
1	0 0 1	1
2	0 1 0	1
3	0 1 1	0
4,6	1 X 0	1
5,7	1 X 1	0

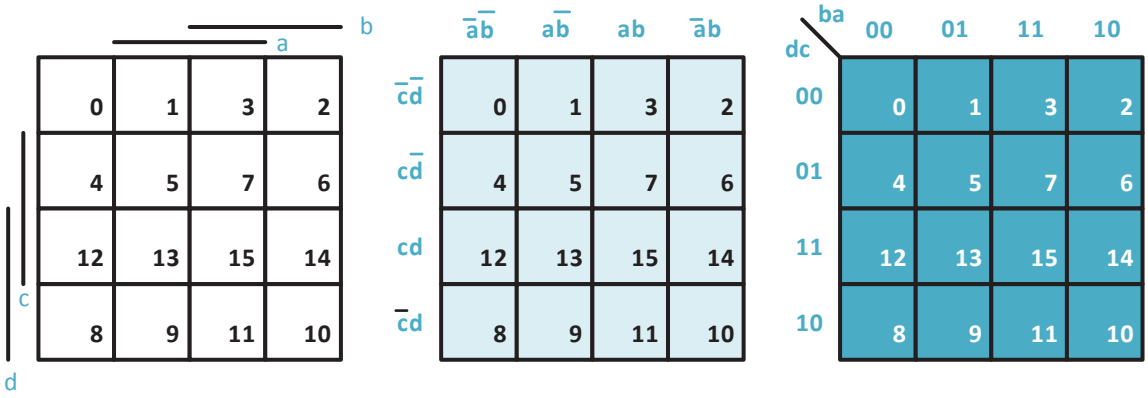
3.1.3 Vennův diagram, Karnaughova mapa

Přehledněji než pomocí pravdivostní tabulky zobrazíme logickou funkci pomocí Vénových diagramů (Obr. 6) nebo pomocí mapy (Obr. 7). Vyšší přehlednost spočívá v tom, že v diagramu i v mapě jsou zobrazeny vedle sebe ty termy, které se liší v jedné proměnné. Jestliže pak je funkční hodnota zkoumané funkce rovna log. 1 v sousedních políčkách mapy nebo diagramu, můžeme vytvořit jednu společnou oblast, která zahrnuje oba termy, výsledný logický výraz je pak jednodušší.



Obr. 6: Vennův diagram funkce tří proměnných

Vennův diagram funkce tří proměnných je vyobrazen na Obr. 6. Jedná se o způsob grafického vyjádření příslušnosti prvků do množiny. Je tvořený uzavřenými křivkami, přičemž body uvnitř křivky představují prvky dané množiny a body venku prvky, které do množiny nepatří.



Obr. 7: Karnaughova mapa funkce o čtyřech proměnných

Na Obr. 7 je vyobrazena Karnaughova mapa funkce o čtyřech proměnných. U jednotlivých příkladů (mapy A, B a C) je využito rozdílného způsobu popisu mapy. Jednotlivé vstupní proměnné nabývají hodnotu log. 1 v těch sloupcích resp. řádcích, u kterých je zakreslen pruh (A), proměnná není negována (B), nebo tam kde je sloupec resp. řádek pro danou proměnnou označen 1. V políčkách map jsou uvedeny příslušné stavové indexy funkce.

3.1.4 Minimalizace logických funkcí

V technické praxi se často snažíme, aby logická funkce byla vyjádřena co nejjednodušším logickým výrazem (snaha realizovat funkci co nejmenším počtem logických prvků). Procesu hledání tohoto výrazu říkáme minimalizace logického výrazu.

Nejprve je třeba stanovit kritéria minimalizace. V prvé řadě se snažíme zmenšit počet termů, ze kterých se výraz skládá. Dále je třeba minimalizovat počet vstupních proměnných, které tvoří jeden term, a nakonec se snažíme minimalizovat počet negací ve výsledném logickém výrazu. Minimálního výrazu můžeme dosáhnout buď pomocí algebraických úprav výchozího výrazu, nebo využít sousedních stavů k redukci počtu a velikosti termů. K nalezení sousedních stavů využijeme buď mapu nebo některý formální algoritmus, umožňující počítačové řešení problému (například algoritmus Quine-Mc Cluskey). Výsledkem minimalizace může být logická funkce ve tvaru logického součtu jednotlivých součinů tzv. MNDF (Minimální Normální Disjunktivní Forma), nebo součinu jednotlivých součtů tzv. MNKF (Minimální Normální Konjunktivní Forma). Tyto tvary logických výrazů pak mohou být dále upravovány na tvar realizovatelný pomocí základních logických obvodů (Tab. 5).

Úprava logického výrazu

Zjednodušte logický výraz pro funkci: $f(d,c,b,a) = \overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}(c + d) + \overline{b}\cdot\overline{c}\cdot\overline{d}$. Využijte algebraických úprav (zákonů Booleovy algebry).

Řešení: Postupnou aplikací zákonů Booleovy algebry upravíme daný výraz na minimální formu.

Výchozí tvar	$\overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}(c + d) + \overline{b}\cdot\overline{c}\cdot\overline{d}$
Distributivní zákon	$\overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}\cdot c + a\cdot\overline{b}\cdot d + \overline{b}\cdot\overline{c}\cdot\overline{d}$
Zákon absorbce negace	$\{\overline{a}\cdot d + a\cdot\overline{b}\cdot d = d(\overline{a}+\overline{b})\}$
	$\overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}\cdot c + \overline{b}\cdot d + \overline{b}\cdot\overline{c}\cdot\overline{d}$
Absorbce negace	$\{\overline{b}\cdot d + \overline{b}\cdot\overline{c}\cdot\overline{d} = \overline{b}(d + \overline{c})\}$
	$\overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}\cdot c + \overline{b}\cdot d + \overline{b}\cdot\overline{c}$
Absorbce negace	$\{a\cdot\overline{b}\cdot c + \overline{b}\cdot\overline{c} = \overline{b}(\overline{c} + a)\}$
	$\overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b} + \overline{b}\cdot d + \overline{b}\cdot\overline{c}$
Absorbce	$\overline{a}\cdot d + \overline{b}\cdot c + a\cdot\overline{b} + \overline{b}\cdot d + \overline{b}\cdot\overline{c}$
Consensus	$f = \overline{a}\cdot d + a\cdot\overline{b} + \overline{b}\cdot\overline{c}$

Výsledný logický výraz funkce f je minimální. Logický výraz, který jsme získali po aplikaci zákona o consensu, je minimálním vyjádřením funkce f.

Minimalizace log. výrazu pomocí Karnaughovy mapy

Minimalizujte logickou funkci f danou výčtem jedničkových stavů:

$f = \sum(1,4,5,6,9,10,12,13,14)$.

Řešení: K minimalizaci použijeme Karnaughovu mapu, na které vyznačíme jedničkové stavy. Účelem použití mapy je nalezení sousedních stavů, tj. stavů, které se liší v jednom bitu. K

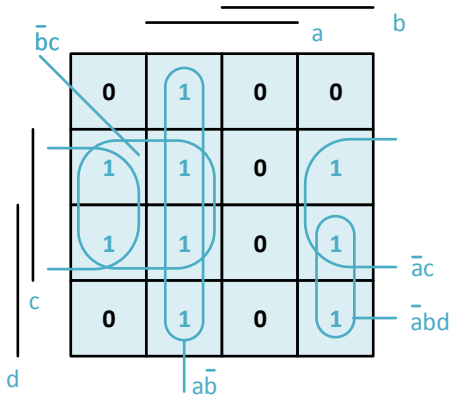
libovolnému políčku mapy je sousední políčko to, které je vedle, pod nebo nad tímto políčkem a všechna políčka, která jsou ve stejném sloupci resp. řádku na opačném konci, jestliže uvažované políčko je na kraji tabulky.

Např. na Obr. 7 jsou k políčku 13 sousední políčka 5, 15, 9, a 12, k políčku 0 sousední políčka 1, 4, 8 a 2. Dále hledáme sousední dvojice políček k vybrané dvojici sousedních políček. Např. ke dvojici políček 0,8 je sousední dvojice 1,9 nebo 2,10 nebo 4,12. Ke každé sousední čtveřici je možné podobným postupem hledat sousední osmici atd. Smyslem tohoto hledání je nalezení dvojic, čtveřic, osmic, … políček, které obsahují funkční hodnotu 1. Při minimalizaci mají přednost větší skupiny sousedů, protože je možné je popsat jednodušším výrazem. V našem příkladu jsme označili čtveřice políček (Obr. 8), které je možno popsat výrazy $\bar{a}\cdot c$, $a\cdot\bar{b}$, $\bar{b}\cdot c$ a dvojici $\bar{a}\cdot b\cdot d$. Označené oblasti se mohou překrývat, všechny jedničky v mapě je však třeba pokrýt alespoň jednou oblastí. V případě, že pro některou jedničku není možné najít žádnou dvojici, je třeba příslušný výraz, popisující dané políčko, do výsledného vyjádření funkce zahrnout. V případě, že se v mapě vyskytují funkční hodnoty X, je možné příslušné políčko využít k vytvoření nějaké oblasti, nebo ve výsledném vyjádření funkce nemusí být zahrnuto. Výsledný tvar funkce je:

$f = \bar{a}\cdot c + a\cdot\bar{b} + \bar{b}\cdot c + \bar{a}\cdot b\cdot d.$

Jestliže chceme výsledný výraz realizovat pouze s pomocí obvodů NAND, musíme dále použít De Morganovo pravidlo k další úpravě výsledného logického výrazu. Po úpravách dostaneme:

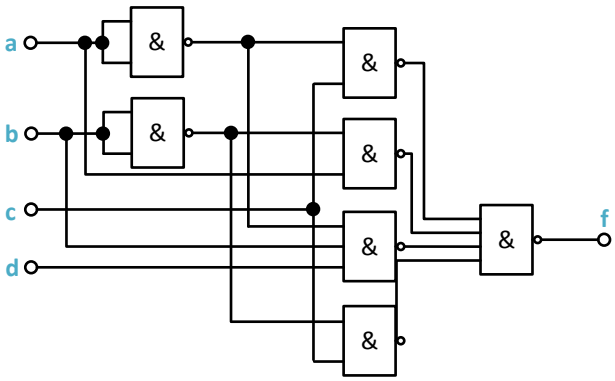
$$f = \overline{\overline{\bar{a}\cdot c} + \overline{a\cdot\bar{b}} + \overline{\bar{b}\cdot c} + \overline{\bar{a}\cdot b\cdot d}}$$



Obr. 8: Minimalizace do MNDF

Logické schéma vytváříme postupně. Pomocí invertorů nebo hradel NAND zapojených tak, aby realizovaly funkci invertoru, zajistíme negaci jednotlivých proměnných. (V uvedeném příkladu jsou to proměnné $\bar{a}$, $\bar{b}$).Pomocí členů NAND s vhodným počtem vstupů realizujeme vnitřní negované logické součiny. (V příkladu jsou to negované součiny $\bar{a}c$, $a\bar{b}$, $\bar{a}bd$):

Poslední výstupní člen NAND s vhodným počtem vstupů provede negovaný součin členů



Obr. 9: Zapojení obvodu realizujícího funkci f.

vytvořených podle předchozího bodu. Výsledné logické schéma je na Obr. 9.

Úloha minimalizace není jednoznačná. V uvedeném příkladu bylo například možno místo pokrývání jedniček v Karnaughově mapě pokrývat nuly a výslednou proměnnou invertovat.

V praxi je návrhář zpravidla omezen ještě dalšími faktory, jako je maximální počet logických úrovní, kterými se v obvodu může signál šířit, maximálním možným počtem vstupů jednotlivých hradel, maximálním možným větvením signálu atd. S výhodou naopak v některých případech může využít tzv. skupinové minimalizace, kdy pro větší počet výstupů využije společnou část obvodu. V současnosti se pro návrh obvodů používá návrhových systémů, které automaticky vygenerují vhodnou vnitřní strukturu zapojení obvodů.

Příklady – minimalizace pomocí mapy

Použijte Karnaughovu mapu pro minimalizaci Booleových výrazů následujících funkcí zadaných logickým výrazem a pravdivostní tabulkou:

$X = \overline{a}\cdot\overline{b} + a\cdot\overline{b}\cdot\overline{c} + a\cdot\overline{b}\cdot c + a\cdot b\cdot c$

$Y = \overline{a}\cdot\overline{b}\cdot\overline{c} + \overline{a}\cdot b\cdot\overline{c}\cdot d + a\cdot\overline{c}\cdot\overline{d} + a\cdot\overline{b}\cdot c\cdot\overline{d}$

Logická funkce je zadána v pravdivostní tabulce:

Tab. 11: Zadání logické funkce

a	b	c	d	Z
0	0	0	0	1
0	0	0	1	0
0	0	1	0	X
0	0	1	1	0
0	1	0	0	1
0	1	0	1	X
0	1	1	0	1
0	1	1	1	1
1	0	0	0	X
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	X

3.1.5 Minimalizace logických funkcí metodou Quine–McCluskey

Minimalizace logických funkcí pomocí Karnaughových map (KM) je relativně rychlá a jednoduchá metoda, kterou lze provádět ručně na papíře. Tato metoda funguje velmi dobře pro 4 a méně proměnných. Minimalizaci pomocí KM lze použít i pro 5 proměnných, ale je méně přehledná a snadno se v ní udělá chyba. Alternativou, která je ekvivalentem minimalizace pomocí KM a umožňuje minimalizaci funkce o více než 5-ti proměnných je využití Quine–McCluskey (QM) algoritmu (někdy nazýván metoda přímých implikantů). QM je metoda určená pro minimalizaci booleovských funkcí, která byla vyvinuta W. V. Quinem a E. J. McCluskym v roce 1956. Tento algoritmus je prováděn pomocí tabulek, může tedy být prováděn ručně na papír, ale je také

vhodný pro algoritmické zpracování pomocí počítače.

V následujícím textu si minimalizaci pomocí QM algoritmu vysvětlíme na jednoduchém příkladu.

Zadaní: Nalezněte MNDF následující funkce

$f(d,c,b,a)=\sum(1,4,7,8,9,10,11,12,14,15)$

Nyní je třeba sestavit tabulky, s jejíž pomocí budeme zadanou funkci minimalizovat. V následujících krocích (a - i) je popsán postup pro sestavení těchto tabulek.

a) Jako první si vytvoříme pravdivostní tabulku zadané logické funkce s tím, že vynecháme ty řádky, kde je hodnota funkce nulová.

Tab. 12: Quine-McCluskey, krok a

index	d	c	b	a
1	0	0	0	1
4	0	1	0	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
14	1	1	1	0
15	1	1	1	1

b) Jako další si vytvoříme tabulku, kde rozdělíme jednotlivé řádky do skupin podle počtu jedniček.

Tab. 13: Quine-McCluskey, krok b

Počet jedniček	index	d	c	b	a
1	1*	0	0	0	1
	4*	0	1	0	0
	8*	1	0	0	0
2	9*	1	0	0	1
	10*	1	0	1	0
	12*	1	1	0	0
3	7*	0	1	1	1
	11*	1	0	1	1
	14*	1	1	1	0
4	15*	1	1	1	1

c) Sestavíme tabulku, kde se pokusíme spojit dvojice řádků, které se liší pouze v jednom bitu, tento bit nahradíme znakem "-„. To které dvojice jsme při porovnání použili, si v předchozí tabulce vyznačíme *.

Tab. 14: Quine-McCluskey, krok c

index	dcba
1-9	-001
4-12	-100
8-9*	100-
8-10*	10-0
8-12*	1-00
9-11*	10-1
10-11*	101-
10-14*	1-10
12-14*	11-0
7-15	-111
11-15*	1-11
14-15*	111-

d) Nyní budeme stejný postup aplikovat na novou tabulku, znak – je třeba chápat jako novou hodnotu. Použité řádky si opět označíme *. (Při porovnávání stačí nalézt řádky se znakem – na stejné pozici). V případě duplicity některé řádky vyškrtneme.

Tab. 15: Quine-McCluskey, krok d

index	dcba
8-9 10-11	10--
~~8-10 9-11~~	~~10--~~
8-10 12-14	1--0
~~8-12 10-14~~	~~1--0~~
10-11 14-15	1-1-
~~10-14 11-15~~	~~1-1-~~

e) Do další tabulky zapíšeme řádky, které jsme nepoužili při žádném porovnání (řádky, které nejsou označeny *) a řádky, které nám zbyly v předchozí tabulce. A vytvoříme tzv. tabulku pokrytí.

Tab. 16: Quine-McCluskey, krok e

index	1	4	7	8	9	10	11	12	14	15
1-9	X				X					
4-12		X						X		
7-15			X							X
8-9 10-11				X	X	X	X			
8-10 12-14				X		X		X	X	
10-11 14-15						X	X		X	X

f) V tabulce pokrytí najdeme ty sloupce, ve kterých je označené jenom jedno políčko (ty to sloupce se označují jako tzv. nesporné implikanty) a tyto sloupce škrtneme. Od těchto sloupců škrtneme celý řádek, a pokud při tom škrtneme další označená políčka, tak škrtneme i další sloupce (ty, které daná políčka obsahují). (škrtané řádky označíme červeně, od nich škrtané sloupce žlutě)

Tab. 17: Quine-McCluskey, krok f

index	výraz	1	4	7	8	9	10	11	12	14	15
1-9	$a{\cdot}b{\cdot}c$	X				X					
4-12	$\bar{a}{\cdot}\bar{b}{\cdot}c$		X						X		
7-15	$a{\cdot}\bar{b}{\cdot}\bar{c}$			X							X
8-9 10-11	$\bar{c}{\cdot}d$				X	X	X	X			
8-10 12-14	$\bar{a}{\cdot}d$				X		X		X	X	
10-11 14-15	$b{\cdot}d$						X	X		X	X

g) Získáme tabulku, kde ve sloupcích jsou všechny jedničkové stavy (tedy stavy, které musíme pokrýt) a v řádcích jsou všechny přímé implikanty (tedy všechny skupiny sousedních stavů)

Tab. 18: Quine-McCluskey, krok f

index	výraz	8	10	11	14
8-9 10-11	$b{\cdot}d$		X	X	X
8-10 12-14	$\bar{a}{\cdot}d$	X	X		X
10-11 14-15	$\bar{c}{\cdot}d$	X	X	X	

h) Vyškrtnutí sloupce, který má hvězdičku ve stejných řádcích jako jiný sloupec nebo má nějaké hvězdičky navíc (dominance sloupce), v příkladu 10. Sloupec. Nebo vyškrtnutí řádku, který je podmnožinou jiného řádku (dominance řádku), v tomto příkladu se nevyskytuje.

Tab. 19: Quine-McCluskey, krok h

index	výraz	8	11	14
8-9 10-11	$b{\cdot}d$		X	X
8-10 12-14	$\bar{a}{\cdot}d$	X		X
10-11 14-15	$\bar{c}{\cdot}d$	X	X	

i) Sestavení výsledného logického výrazu. Použijeme tří přímých implikantů získaných v bodě f, zbylé termy vybereme z předešlé tabulky dle kritérií minimality (minimální počet termů, minimální počet nezávislých proměnných v každém termu, minimální počet negovaných proměnných v každém termu).

Výsledek:

$f(d,c,b,a) = a{\cdot}\bar{b}{\cdot}\bar{c} + \bar{a}{\cdot}\bar{b}{\cdot}c + a{\cdot}b{\cdot}c + b{\cdot}d + \bar{a}{\cdot}d$

nebo:

$f(d,c,b,a) = a{\cdot}\bar{b}{\cdot}\bar{c} + \bar{a}{\cdot}\bar{b}{\cdot}c + a{\cdot}b{\cdot}c + b{\cdot}d + \bar{c}{\cdot}d$

Dle kritérií minimality byl vybrán term $b{\cdot}d$, který neobsahuje žádnou negaci. Zbylé dva termy mají stejnou délku a obsahují stejný počet negací, můžeme tedy vybrat kterýkoli z nich (proto máme dva výsledky).

3.1.6 Příklady k procvičování:

Pomocí algebraických úprav nalezněte MNDF funkce f: $f = a{\cdot}\bar{b}{\cdot}c + a(b+\bar{c}) + (a+b).(\bar{a}+c)$

Upravte logický výraz vyjadřující funkci f tak, aby tato funkce byla popsána pomocí minimálního počtu hradel NAND: $f = a(\bar{b}+\bar{c}) + a{\cdot}\bar{b}{\cdot}c + a{\cdot}b{\cdot}c{\cdot}d$

Využitím zákonů Booleovy algebry minimalizujte následující logický výraz:

$(a + b + \overline{c})(a + \overline{a}\cdot b + \overline{a}\cdot d)$

Pomocí algebraických úprav nalezněte MNDF funkce f: $f = a(\overline{b} + \overline{c}) + a\cdot\overline{b}\cdot c + a\cdot b\cdot c\cdot d$

Pomocí algebraických úprav nalezněte MNDF funkce f: $f = \overline{a}\cdot d + \overline{b}\cdot c\cdot d + a\cdot\overline{b}(c + d) + \overline{b}\cdot\overline{c}\cdot\overline{d}$

Navrhněte schéma zapojení, které bude realizovat dekodér z BCD kódu do kódu sedmisegmentového displeje. Úlohu řešte pouze pro horní vodorovný segment displeje. Proveďte minimalizaci zapojení. Realizujte pomocí hradel NAND.

Navrhněte schéma zapojení, které bude realizovat dekodér z BCD kódu do kódu sedmisegmentového displeje. Úlohu řešte pouze pro horní vodorovný segment displeje. Proveďte minimalizaci zapojení. Realizujte pomocí hradel NAND

Navrhněte schéma zapojení, které bude pro 4 vstupní proměnné realizovat prahovou funkci s prahem ≥ 11. K realizaci použijte hradel NAND, minimalizaci proveďte pomocí úpravy algebraického výrazu.

Popište prahovou funkci čtyř proměnných s prahem 12 (vyjádřeno dekadicky) pomocí UNDF a UNKF. Oba logické výrazy minimalizujte a převeďte do tvaru, vhodného pro realizaci pomocí logických členů NAND a NOR .

Navrhněte obvod hlídače liché parity 3 bitové informace pomocí hradel NAND.

Navrhněte schéma zapojení, které bude doplňovat paritní bit liché parity ke tříbitové informaci. (vstupní proměnné: a, b, c výstupní proměnná: p). K realizaci použijte hradel NAND, minimalizaci proveďte pomocí úpravy algebraického výrazu.

Navrhněte schéma zapojení, které bude doplňovat majoritní bit ke tříbitové informaci. (vstupní proměnné: a, b, c, výstupní proměnná: p). K realizaci použijte hradel NAND, minimalizaci proveďte pomocí úpravy algebraického výrazu.

Na Karnaughově mapě pro tři proměnné a b c vyznačte sousední pole k poli reprezentujícímu term $\overline{a}\cdot\overline{b}\cdot\overline{c}$.

4 Technologie výroby číslicových obvodů

Číslicové elektronické obvody jsou vyrobeny tak, aby jejich prostřednictvím bylo možno realizovat logické funkce. Snahou výrobců je co nejvíce se přiblížit ideální funkci obvodů, pro kombinační obvody to znamená nastavit na výstupu obvodu co nejdříve výsledek logické operace se vstupními proměnnými. Jednotliví výrobci obvodů značí integrované obvody podle jednotného schématu (viz příloha 4).

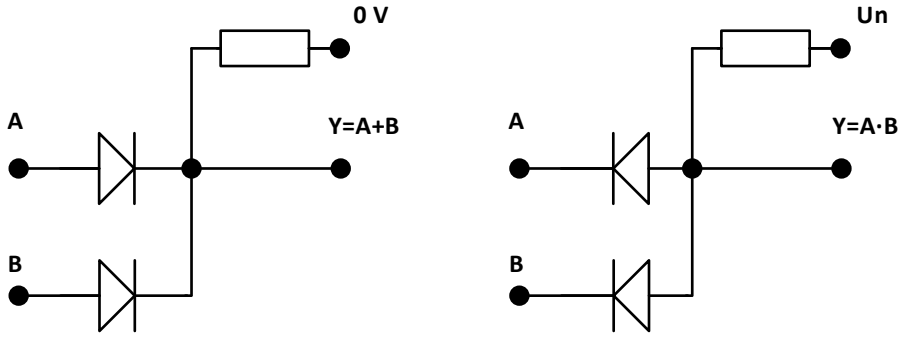
4.1 Zkoumané vlastnosti

Míru přiblížení se k ideálním vlastnostem můžeme zkoumat pomocí různých měřítek. Zpravidla se zajímáme o to, jak daný obvod zatěžuje předchozí obvody, logicky jej napájející (vstupní charakteristika), dále se zajímáme o průběh zpracování vstupního signálů (převodní charakteristika a zpoždění signálu při průchodu obvodem) a o zatížitelnost obvodu dalšími obvody (zatěžovací charakteristika). Důležitými vlastnostmi obvodu je také jeho spotřeba a to jak v klidovém stavu tak při přepínání mezi stavy. Další vlastností je odolnost proti rušení a u složitějších obvodů existence hazardů. Jelikož výrobci zpravidla neudávají přesně všechny charakteristiky, používají se další míry, charakterizující obvody:

- Vstupní větvení udává hypotetický počet standardních vstupů, které by zatěžovaly předchozí výstupy obvodů stejně jako uvažované hradlo.
- Výstupní větvení udává počet standardních vstupů hradel dané technologie, které je možno připojit k výstupu daného hradla.
- Logický zdvih (viz Obr. 2) udává rozdíl napětí pro log. 1 a log. 0. Hodnota logického zdvihu ovlivňuje odolnost proti rušení.
- Zpoždění signálu při průchodu jedním hradlem je nejsledovanějším parametrem číslicových obvodů, neboť tento parametr přímo ovlivňuje rychlost výpočtů prováděných pomocí číslicových obvodů.

4.2 Diodová logika

Pro pochopení principu číslicových obvodů si nejprve uvedeme příklad diodové logiky, která se v moderních obvodech již nepoužívá, ale je velmi názorná. Zkoumejme nejprve vytvoření logického součinu pomocí diodové logiky. Na rozdíl od příkladu uvedeného na Obr. 3 budeme vytvářet logický součin signálů s logickými úrovněmi log. 1 a log. 0 reprezentovaných napěťovými



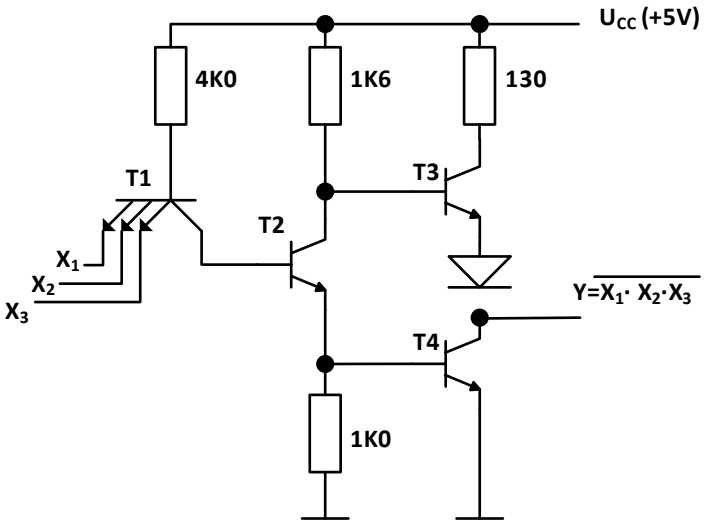
Obr. 10: Příklad diodové logiky, vytvoření logického součtu a logického součinu

úrovněmi Un a 0 V. Obvod realizující funkci log. součinu je na Obr. 10.

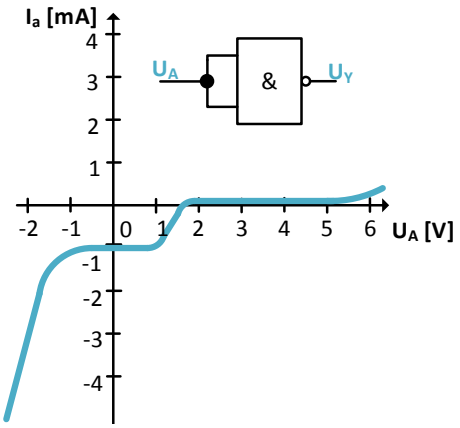
Předpokládejme nejprve, že napětí na vstupech A a B je log. 1, potom na výstupu Y bude napětí rovno Un, což odpovídá log. 1. V případě, že alespoň na jednom ze vstupů bude log. 0, bude na výstupu Y vzhledem k otevření příslušné diody napětí odpovídající log. 0.Obdobně je možno sestrojit obvod realizující logický součet. (Obr. 10). Diodová logika má podstatná omezení. Na odporech vznikají úbytky napětí, které způsobují degradaci signálu a omezují řazení počtu stupňů diodové logiky na max. dva za sebou. Dále v této logice není možno provést negaci signálu. Tyto problémy řeší diodově-tranzistorová logika, která využívá spojení diodové logiky s invertujícím tranzistorovým zesilovačem. Tato logika je však už zastaralá, její dynamické parametry jsou nevyhovující.

4.3 Logika TTL

Hradla TTL používají více emitorového transistoru k vytvoření logické funkce (transistor T1 na Obr. 11), dvojčinného invertujícího stupně s budičem (T2,T3,T4) k úpravě výstupního signálu. Existují různé technologické varianty těchto obvodů: H-rychlá, N-normální, L-nízká spotřeba, S-se Schotkyho diodami (zabraňují saturaci transistorů), LS, ALS nízká spotřeba + Schotkyho diody. Z charakteristik hradel TTL uvádíme vstupní charakteristiku (Obr. 12), převodní charakteristiku (Obr. 13), zatěžovací charakteristiku (Obr. 14) a odběrovou charakteristiku (Obr. 15). Konkrétní průběh charakteristik je závislý na teplotě. Na Obr. 16 je znázorněna základní představa o statické odolnosti obvodu TTL proti rušení na vstupu. Odolnost proti rušení je zajištěna pro rušivé signály

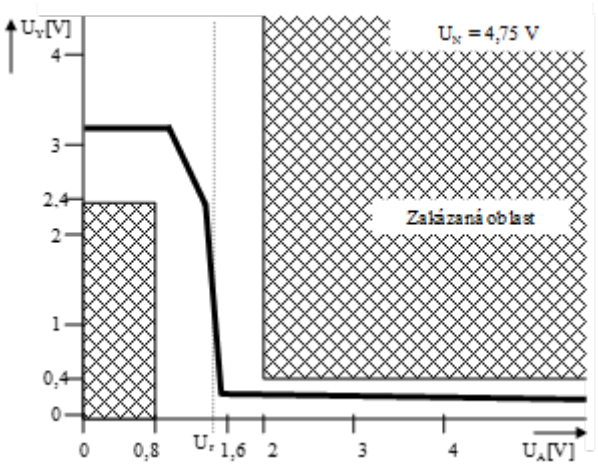


Obr. 11: Třívstupové hradlo NAND v technologii TTL



Obr. 12: Vstupní charakteristika hradla TTL

maximálně do amplitudy 0,4 V. Vyšší amplituda rušivého signálu může způsobit zákmit logické hodnoty na výstupu.



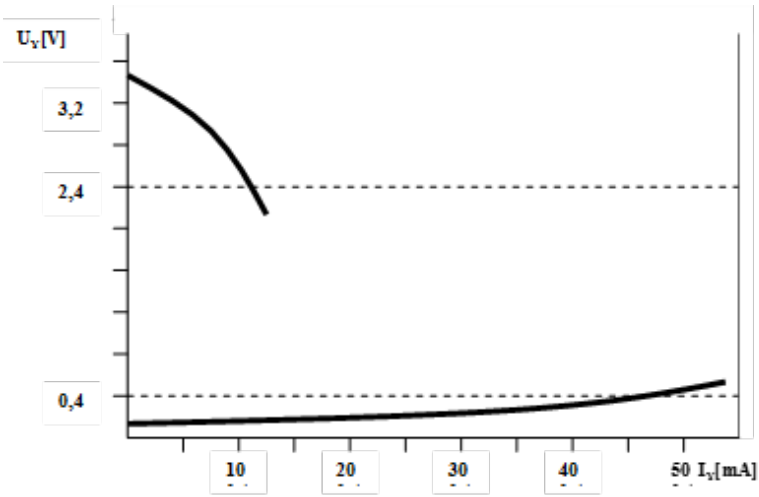
Obr. 13: Převodní charakteristika invertoru TTL

Na Obr. 11 je vyobrazeno třívstupové hradlo NAND v technologii TTL. Je-li některý ze vstupů v log0, T1 otevřen, T2 a T4 uzavřen, T3 otevřen. Jsou-li všechny vstupy v log1, T1, T3 uzavřeny, T2 a T4 otevřeny.

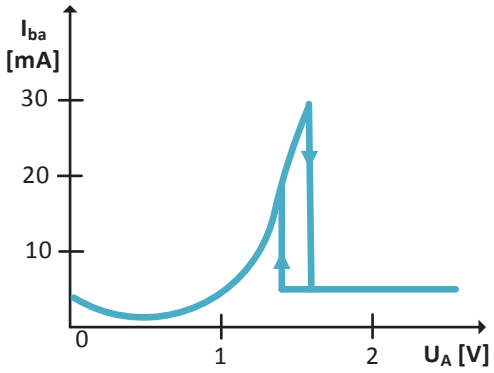
Významným bodem vstupní charakteristiky je hodnota 1,5 V vstupního napětí pro nulový vstupní proud. Tato hodnota se ustálí na nezapojených vstupech hradla.

Napětí UA představuje vstupní napětí, UY představuje výstupní napětí. Šrafováním vyznačeny oblasti, které jsou mimo výrobní toleranci vstupního a výstupního napětí. Podstatné hodnoty napětí jsou:

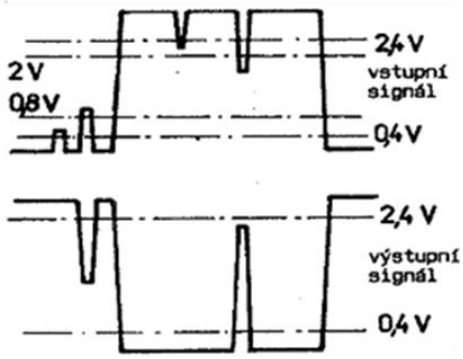
- 0,8 V – max. hodnota UA odpovídající log. 0,
- 2,0 V – min. hodnota UA odpovídající log. 1,
- 2,4 V – max. hodnota UY odpovídající log. 0
- 0,4 V – min. hodnota UY odpovídající log. 1
- Ur – překlápěcí úroveň vstupního napětí



Obr. 14: Zatěžovací charakteristiky hradla TTL pro případ zatěžování výstupu nastaveného do log. 1 a do log. 0.



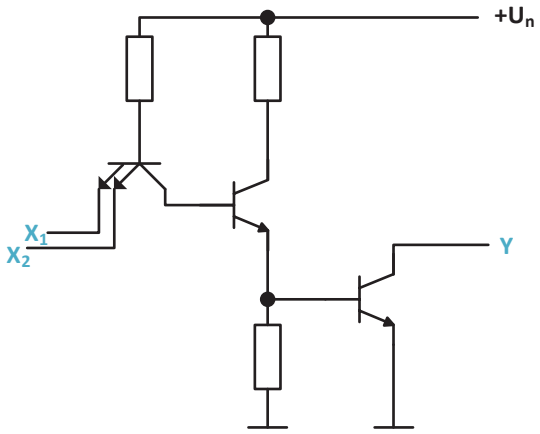
Obr. 15: Odběrová charakteristika invertoru TTL. U_A je vstupní napětí na invertoru, I_{CC} proud odebíraný z napájecího zdroje



Obr. 16: Odolnost proti statickému rušení invetoru TTL

4.4 Obvody s otevřeným kolektorem

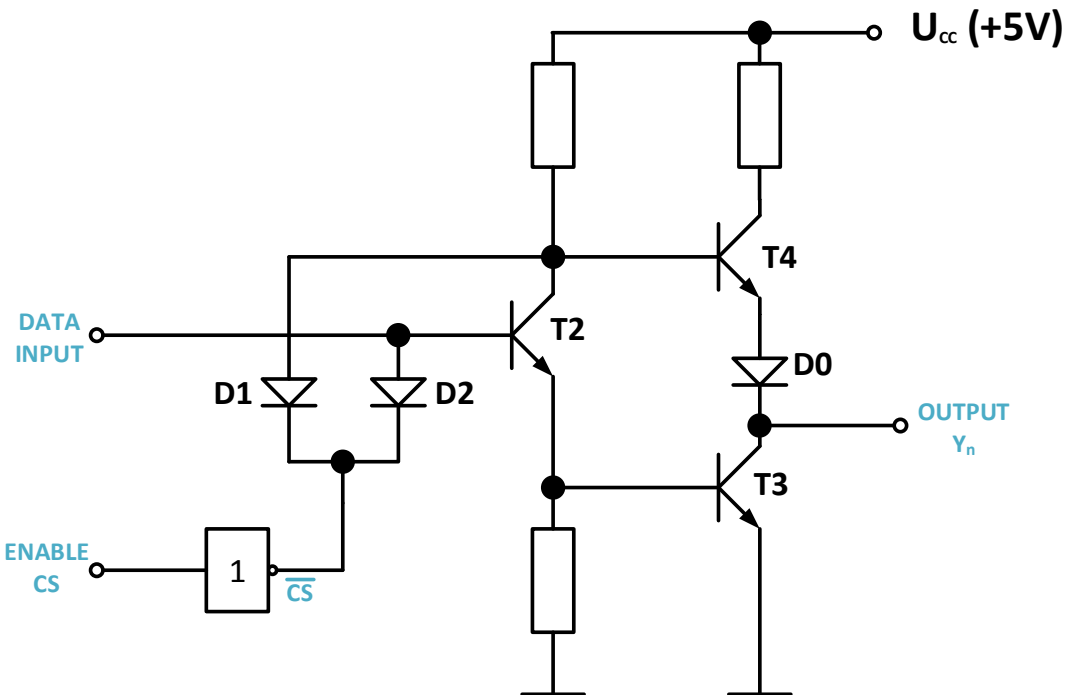
Zapojení TTL hradel s otevřeným kolektorem může být stejné jako u běžného hradla (Obr. 11) s tím rozdílem, že ve schématu je vynechán transistor T3 a odpor 130 Ohmů. Zapojení obvodu s otevřeným kolektorem je vyobrazeno na Obr. 17. Chybějící transistor pak musí být nahrazen externím odporem nebo dalším hradlem. Výhoda obvodů s otevřeným kolektorem spočívá v tom, že umožňuje spojení výstupů několika hradel. Takto spojené výstupy realizují funkci logického součinu, kterému se běžně říká „montážní součin“. Na rozdíl od obvodů s otevřeným kolektorem je spojení výstupů běžných hradel nepřípustné, neboť může dojít k jejich zničení.



Obr. 17: TTL obvod s otevřeným kolektorem

4.5 Hradla s třetím stavem

Třístavová hradla pracují ve dvou základních režimech: v normálním režimu hradlo provádí požadovanou logickou funkci stejně jako hradlo standardní. V režimu vysoké impedance (třetí stav) hradlo vykazuje „nekonečnou“ výstupní impedanci. K řízení těchto dvou režimů se používá pomocného logického vstupu ovládajícího přídavné obvody, které umožňují současně uzavřít transistory T3 a T4 (stav vysoké impedance výstupu). Obvody se využívají při návrhu sběrnic.



Obr. 18: Tří stavové hradlo

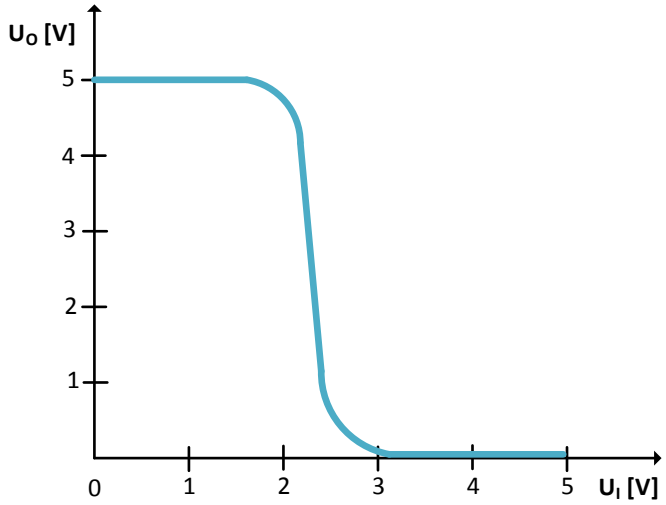
4.6 Ošetření nepoužitých vstupů

Ze vstupní charakteristiky vyplývá, že není vhodné ponechávat u logiky TTL nezapojené vstupy (nezapojený vstup je zpravidla interpretován jako log. 1, napětí, které se ustálí na tomto vstupu, leží v zakázaném pásmu a existuje nebezpečí špatné interpretace tohoto napětí obvodem). U členů NAND je možno nepoužité vstupy paralelně připojit ke vstupům použitým nebo je připojit na úroveň H. U členů NOR je třeba nepoužité vstupy připojit na úroveň L.

4.7 CMOS technologie

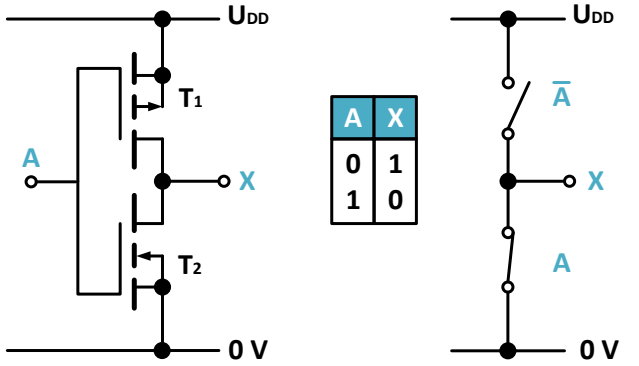
V současnosti je CMOS technologie nejběžnější technologií při realizaci číslicových obvodů. CMOS obvody mají téměř nulovou klidovou spotřebu, odběrová charakteristika však vykazuje relativně velké odběrové maximum při překlápění mezi 0 a 1. Velkou výhodou této technologie je to, že umožňuje vysokou hustotu integrace. Nevýhodou je to, že maximální počet vstupů do jednoho hradla je zpravidla roven 2. Větší počet vstupů do jednoho hradla je nevýhodný z hlediska dynamických vlastností takovéhoto zapojení a bývá nahrazován kaskádou dvouvstupových hradel. Dnešní CMOS obvody jsou již dostatečně rychlé, pro většinu aplikací jsou plnohodnotnou náhradou za obvody TTL. Integrované obvody CMOS jsou označovány jako „unipolární“. Základní jednotku tvoří komplementární tranzistory MOS. Hodnotu napájecího napětí je možno volit v rozsahu od cca 1,5 V do 15 V. Volba napájecího napětí ovlivňuje zejména tyto parametry: rychlost obvodu, šumovou imunitu a spotřebu z napájecího zdroje. U obvodů CMOS je třeba vždy připojit vstupy obvodu na výstup jiného obvodu, na napájecí napětí nebo na zem. Obvody je třeba budit

signály, které mají dostatečně strmé náběžné a sestupné hrany, neboť při pomalejších změnách prudce vzrůstá spotřeba obvodu. Klidová spotřeba je velmi nízká (v každé cestě mezi napájecími vstupy hradla je alespoň jeden z tranzistorů uzavřen). Spotřeba obvodu je tedy úměrná frekvenci změn vstupních signálů. Jestliže obvody nejsou chráněny substrátovými diodami proti přepětí na vstupu, je třeba zabránit vzniku a uplatnění statické elektřiny, která může zničit obvod.

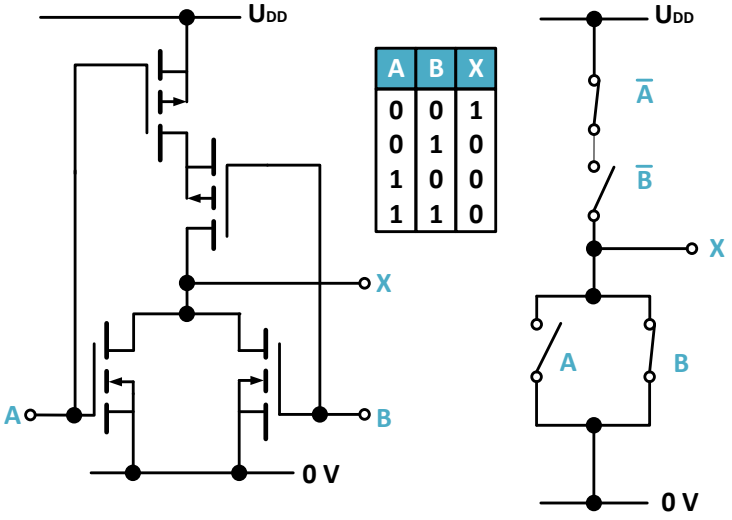


Obr. 19: Převodní charakteristika invertoru CMOS

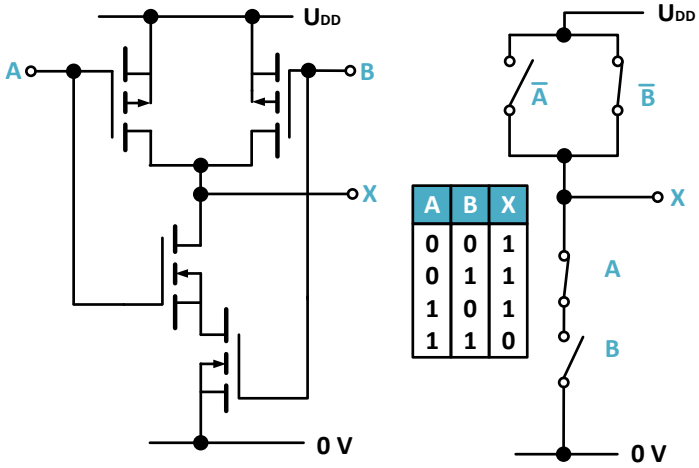
Maximální výstupní větvení je větší než 100, tzn. na jeden výstup hradla je možno připojit více než 100 vstupů dalších hradel. Tato možnost je dána tím, že vstupní proud do tranzistoru FET je zanedbatelný. Velká vstupní kapacita obvodů ovšem zhoršuje dynamické vlastnosti.



Obr. 20: Invertor v CMOS technologii



Obr. 21: Hradlo NOR v CMOS technologii



Obr. 22: Hradlo NAND v CMOS technologii

Na dalších obrázcích jsou vyobrazena hradla vyrobená technologií CMOS. Na Obr. 20 představuje invertor, kde při přivedení napětí UDD na oba transistory se T1 otevře, T2 zůstane uzavřen. Při přivedení napětí 0V se otevře T2 a uzavře T1. Na Obr. 21 můžeme vidět hradlo NOR a na Obr 22 hradlo NAND.

4.8 Další technologie výroby integrovaných obvodů

V současné době prochází technologie výroby IO rychlým vývojem, o čemž svědčí prudké zvyšování výkonu výpočetní techniky. Nejrychlejší obvody jsou realizovány v současnosti technologií ECL (zpoždění signálu při průchodu hradlem < 1 nanosekunda), probíhají pokusy s vytvářením rychlých obvodů v supravodivých materiálech (zpoždění v řádu pikosekund).

Chybí tu např. BiCMOS, FinFET aj.

4.9 Slučitelnost jednotlivých technologií

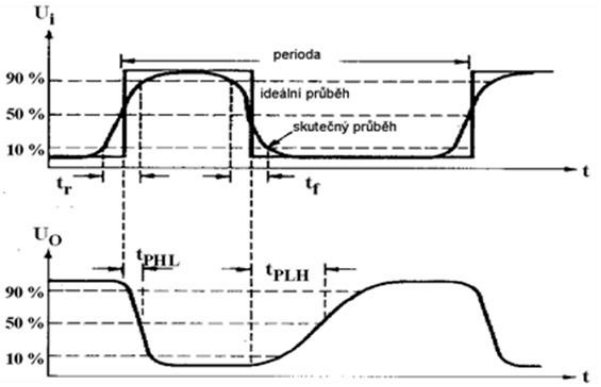
Obecně můžeme říci, že není možno kombinovat v jednom zařízení různé technologie bez přizpůsobení signálů. Např. při porovnání převodní charakteristiky TTL a CMOS vidíme, že rozhodovací úrovně obou těchto technologií jsou vůči sobě navzájem posunuty tak, že obvody CMOS interpretují log. 1 TTL jako log. 0. Proto používáme obvody různých technologií členy, které upravují příslušné úrovně napětí a impedance.

Tab. 20: Srovnání základních parametrů různých technologií výroby číslicových obvodů

Technologie	TTL			CMOS		ECL	
Parametry	74 LS	74 AS	74 ALS	74 C	74 HC	10K	100K
Napájecí napětí,V	5	5	5	5	5	-5,2	-4,5
Maximální úroveň log. 0 na výstupu, V	0,5	0,5	0,5	0,4	0,4	-1,7	-1,7
Minimální úroveň log. 1 na výstupu, V	2,7	2,7	2,7	4,2	4,2	-0,9	-0,9
Minimální úroveň log. 1 na vstupu, V	2,0	2,0	2,0	3,5	3,5	-1,2	-1,2
Maximální úroveň log. 0 na vstupu, V	0,8	0,8	0,8	1,0	1,0	-1,4	-1,4
Logický zdvih, V	2,0	2,0	2,0	3,8	3,8	0,8	0,8
Ztrátový výkon na hradlo, mW	2	20	1	0	0	24	40
Zpoždění signálu, ns	10	1,5	4	30	10	2	0,75
Výstupní větvení	10	10	10	>100	>100	10	10

Parametry pro CMOS technologie byly získány pro zatěžovací proudy 4 mA (log. 0) a 2 mA (log. 1). V Tab. 10 vidíme, že k jednotlivým výstupům hradel můžeme připojit pouze omezený počet dalších vstupů. Jelikož toto omezení není možno vždy dodržet, vytvářejí se tzv. budiče. Tyto obvody nevykonávají žádnou logickou funkci, slouží pouze jako neinvertující zesilovače signálu, tzn., zvyšují výstupní větvení.

Návrhář musí zvolit takovou technologii používaných obvodů, která zaručí, že prahové úrovně napětí a zaručené logické úrovně jsou dostatečně od sebe vzdáleny tak, aby žádný z uvedených šumů nezpůsobil nežádoucí změnu stavu zařízení.



Obr. 23: Zpoždění signálu při průchodu hradlem

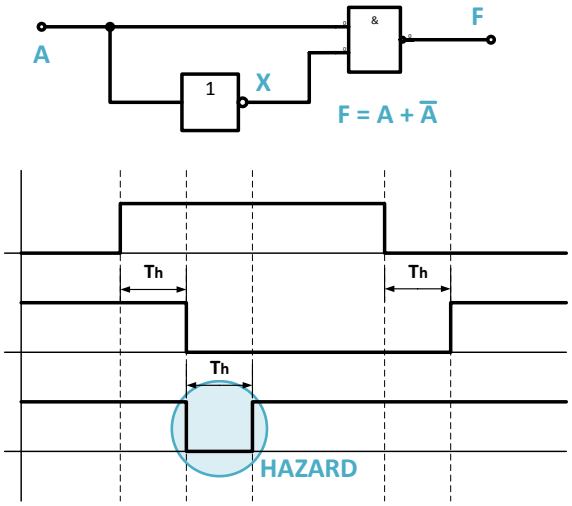
4.10 Rušení v číslicových systémech

Velkou výhodou číslicových systémů ve srovnání s analogovou technikou je velká tolerance vůči šumu. Odolnost jednotlivých komponent systému je základním faktorem, který ovlivňuje spolehlivost systému. Šum může pocházet v číslicovém systému z více zdrojů: Termální (Johnsonův) šum je způsoben rezistory, výstřelový (shot) šum je způsoben náhodným přenesením náboje mezi transistory, 1/f šum je zpravidla způsoben náhodnými variacemi rozšíření nosičů náboje mezi jednotlivými částmi zařízení, interferenční šum souvisí s přenosem náboje od blízkých zařízení kapacitní, nebo indukční vazbou. Při konstrukci obvodu je třeba dbát na to, aby celkový šum byl tak nízký, že neovlivní správnou funkci obvodu.

4.11 Zpoždění a hazardy

Obr. 23 ukazuje typickou závislost průběhu výstupního signálu na vstupním při periodické změně vstupního signálu. Na obrázku si můžeme všimnout, že skutečný průběh signálu není obdélníkový, při průchodu signálu hradlem dochází ještě k jeho další deformaci a že reakce obvodu na změnu vstupu je zpožděná. Průměrná hodnota zpoždění se udává jako veličina tp, kde tp =1/2(tpLH+tpHL). Různě velká zpoždění signálu, který se šíří více rekonvergentními cestami v obvodu může způsobit hazardy, tj. krátké zákmity výstupní logické hodnoty do stavu opačného než je předpokládaná výstupní logická hodnota. Příklad jednoduchého obvodu s hazardem je na Obr. 24.

Na Obr. 24 vidíme časovou analýzu hazardu v chování obvodu realizujícím funkci F. Ideální funkce obvodu je konstantní (obvod by bylo možno nahradit signálem log. 1). Reálný časový průběh signálu F má krátký zákmit do log. 0 způsobený nestejným zpožděním signálů v jednotlivých větvích obvodu. Tento zákmit nazýváme hazardem.



Obr. 24: Časová analýza hazardu

5 Realizace logických funkcí kombinačními obvody

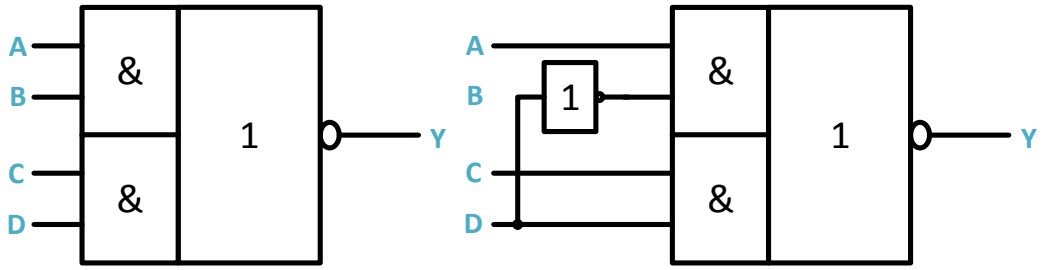
Kombinační funkce můžeme realizovat pomocí základních hradel (NAND, NOR, XOR, NOT,…). V jednotlivých technologiích byly vytvořeny řady základních obvodů, tak aby bylo s jejich pomocí možno snadno realizovat logické obvody. Druhou možností je využít k vytvoření vlastní logické funkce standardních obvodů s vyššími funkčními celky. Tento způsob je výhodný zvláště u složitějších funkcí, kde bychom obtížně sestavovali každý obvod „od začátku" ze základních hradel. Třetí možností je využít programovatelných logických obvodů, o této možnosti bude pojednáno v samostatné kapitole.

5.1 Realizace obvodu pomocí základních hradel

Vzhledem k tomu, že pomocí obvodů NAND můžeme realizovat libovolnou logickou funkci, je jako základní hradlo použit zpravidla 2-8 vstupový obvod NAND pro technologii TTL, 2 vstupový NAND pro technologii CMOS. V tomto případě se více vstupová hradla nahrazují kaskádou dvouvstupových hradel. Existuje více požadavků, které jsou kladeny na výsledný tvar výrazu. Při návrhu obvodů je třeba také uvažovat omezení dané maximálním počtem hradel, kterými se může signál v obvodu šířit (existuje věta, která říká, že každý logický výraz je možno upravit tak, aby počet logických úrovní byl menší nebo roven třem), někdy je třeba zabránit možnosti vytváření hazardů. U složitějších obvodů je třeba dbát na to, aby navržený obvod byl snadno testovatelný.

5.2 Realizace složitějších obvodu pomocí standardních funkčních celků

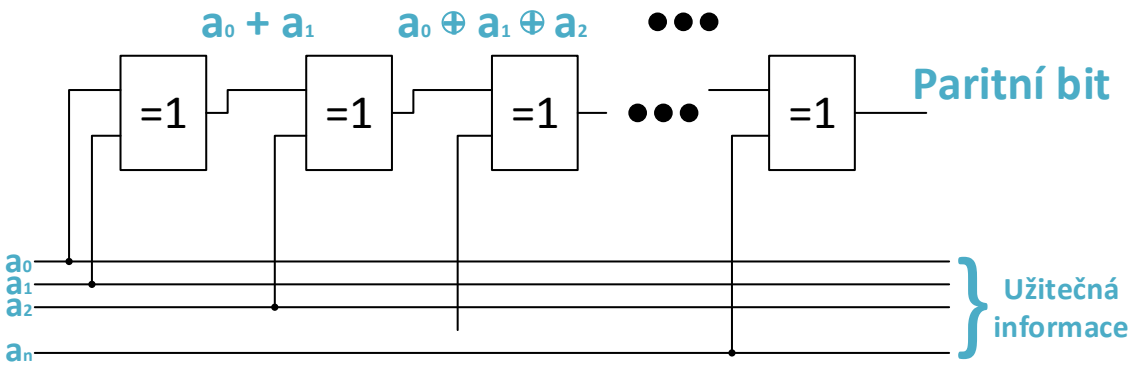
V návrhu kombinačních obvodů se často opakují některá zapojení. Ukázalo se, že je vhodné tato zapojení předpřipravit ve formě samostatného integrovaného obvodu a ušetřit tak návrháři práci s jejich návrhem. Jedním z těchto zapojení je AND-OR-INVERT (Obr. 25). Používá se s výhodou jako základní konstrukční zapojení v zákaznických obvodech v technologii CMOS.Funkce obvodu je zřejmá ze schematické značky: vytvoří se log. součin ze vstupů A a B a současně ze vstupů C a D, z těchto součinů se vytvoří invertovaný součet. Tento obvod je možno využít pro konstrukci logické výhybky. Jednobitová výhybka umožňuje výběr mezi signály A a B pomocí řídícího vstupu V, na výstupu Y se objeví ta informace, která je přivedena na vybraný vstup. Vícebitová výhybka může být vytvořena z jednobitových výhybek tak, že řídící vstup V je společný pro všechny jednobitové výhybky.



Schématická značka Jednobitová výhybka

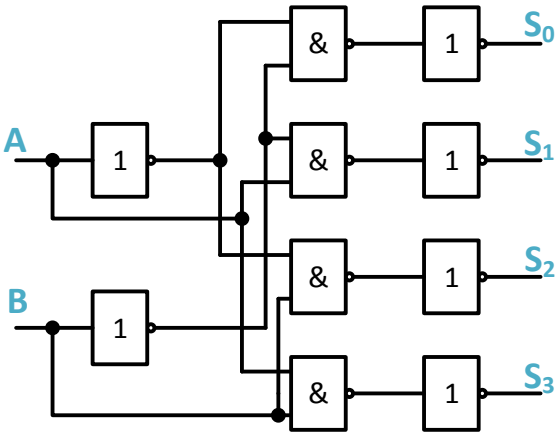
Obr. 25: Obvod AND-OR-INVERT. schématická značka a příklad použití pro konstrukci jednobitové výhybky

Užitečným obvodem je obvod generování resp. kontroly parity (Obr. 26). Parita může být sudá nebo lichá. Jestliže počet jedniček ve slově, které obsahuje informační a paritní bity je lichý, jedná se o paritu lichou, v opačném případě o paritu sudou. Obr. 26 můžeme chápat jednak jako generátor sudé parity, jednak jako obvod kontroly parity, kdy na výstupním bitu bude hodnota 1 v případě, že parita užitečné informace je lichá.



Obr. 26: Generování sudé parity z n bitové informace

Důležitými obvody jsou kodéry a dekodéry. Pod pojmem kodér rozumíme obvod, který převádí informaci z méně komprimovaných kódů (např. kód 1 z N) do kódů komprimovanějších např. do kódu binárního. Obvody provádějící opačnou funkci říkáme dekodéry. Nejčastěji převádíme informaci z binárního kódu nebo z kódu BCD do kódu 1 z N. Příklad dekodéru z binárního dvoubitového kódu do kódu 1 ze 4 je na Obr. 27. V praxi se ovšem používají dekodéry složitější. Například dekodér 7442, což je dekodér z kódu BCD do kódu 1 z 10 a dekodér 74154 což je dekodér z binárního kódu do kódu 1 ze 16 (Obr. 28).

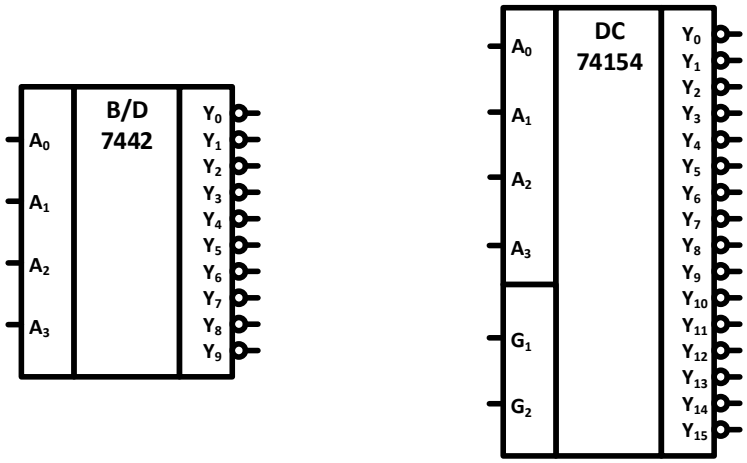


Obr. 27: Principiální schéma dekodéru

Na Obr. 28 můžeme vidět schematické značky dekodéru z kódu BCD do kódu 1 z 10 (obvod číslo 7442) a dekodéru z binárního 4 bitového kódu do kódu 1 ze 16 (obvod číslo 74154). Vstupy G1 a G2 u obvodu 74154 umožňují rozšíření funkce dekodéru na více vstupních proměnných propojením více shodných obvodů. Je-li G1 a současně G2 rovno 0, dekodér nastavuje výstupy podle odpovídající pravdivostní tabulky.

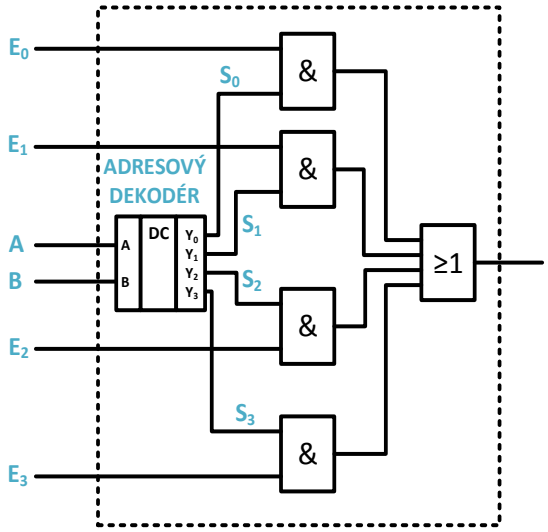
Multiplexor je ve své podstatě výhybkou, která přepíná více než dva vstupy. Principiálně si můžeme jeho činnost popsat pomocí Obr. 29. Skládá se z adresového dekodéru, který převede binární informaci o adrese do kódu 1 ze 4 a z hradlovacích obvodů, které uvolní cestu na výstup pouze tomu vstupnímu signálu, jehož adresa je vybrána.

Vnitřní schéma zapojení multiplexoru pomocí hradel (bez použití adresového dekodéru) je vyobrazeno na Obr. 30. V praxi ovšem používáme multiplexory s větším počtem adres (vstupů). Obecnou schématickou značku multiplexoru můžeme vidět na Obr. 31, příklad multiplexoru

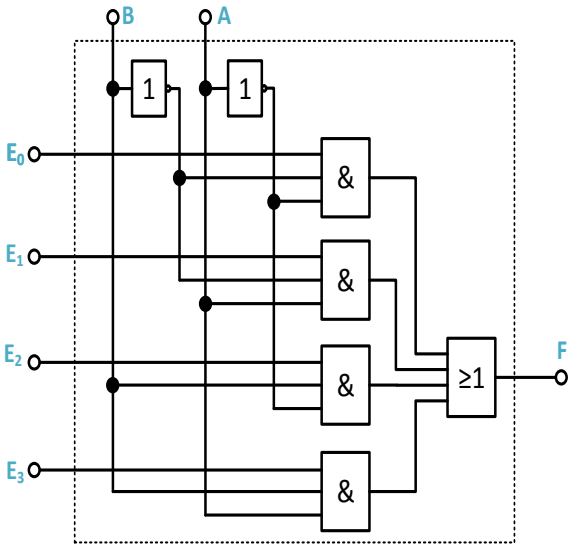


Obr. 28: Schématické značky dekodéru 7442 a 74154

74151 je na Obr. 32. Je-li S (Select) nastaven do log. 1, obvod není vybrán a na výstupu Q je nastavena konstantní hodnota, je-li S=0, obvod pracuje podle pravdivostní tabulky multiplexoru. Název multiplexor se používá i pro složitější obvody, které přepínají mezi více vícebitovými vstupy.



Obr. 29: Principiální schéma multiplexoru



Obr. 30: Skutečné schema multiplexoru

5.3 Realizace logické funkce pomocí dekodéru a multiplexoru

Standardními funkčními bloky je možno realizovat různé logické funkce. Možnosti využití dekodéru a multiplexoru pro realizaci obvodu realizujícího danou funkci budeme demonstrovat na příkladu.

Př. 2: Alarm

Realizujte obvod spuštění alarmu pro zabezpečenou místnost se 4 senzory: T a U - tlakové senzory, V optický senzor, X - infračervený senzor. Alarm se aktivuje (alarm aktivní pro log. 1), jestliže alespoň dva ze senzorů T, U, V a X jsou aktivovány (na jejich výstupu je log. 1). Nyní si ukážeme řešení této úlohy s využitím hradel NAND, dekodéru a multiplexoru. Zadání úlohy odpovídá funkci majority ze čtyř proměnných, kdy připouštíme, že i polovina jedniček ve vstupním slově tvoří majoritu. Funkci

majority můžeme popsat pomocí následující pravdivostní tabulky.

Tab. 21: Pravdivostní tabulka majoritní funkce 4 proměnných

index	t	u	v	x	s
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

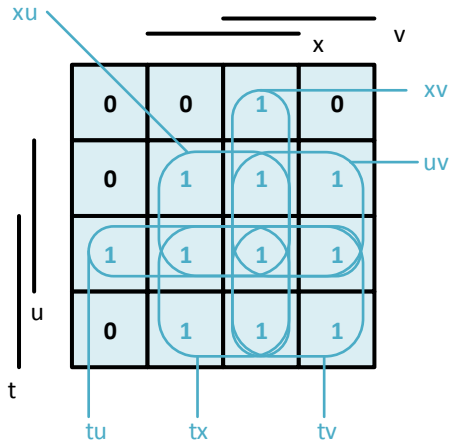
Pro realizaci hradly NAND je zapotřebí z pravdivostní tabulky získat minimalizovanou logickou funkci, kterou je možné po úpravě (pomocí De Moraganovo pravidlo) zapojit pomocí hradel NAND. Pro získání minimální logické funkce využijeme Karnaughovy mapy. Karnaughova mapa pro minimalizaci Tab. 11 je vyobrazena na Obr. 33. Vyčtený minimální logický výraz (MNDF) dále upravíme pomocí De Morganova pravidla.

MNDF:

$s=vx+ux+uv+tx+tv+tu$

Úprava pomoví De Morganova pravidla:

$s=\overline{(vx+ux+uv+tx+tv+tu)}$



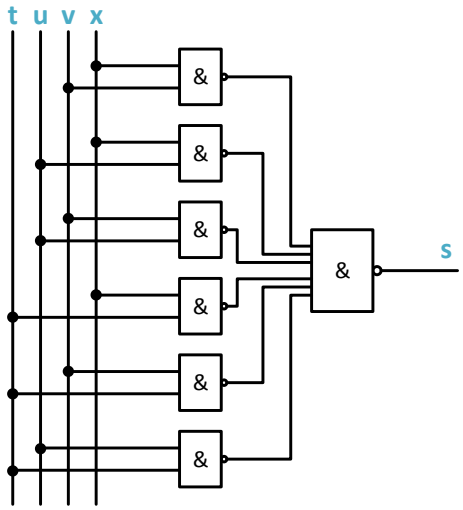
Obr. 31: Karnaughova mapa pro minimalizaci dané úlohy

Výraz upravený pro realizaci hradly NAND:

$$s = \overline{\overline{vx} \cdot \overline{ux} \cdot \overline{uv} \cdot \overline{tx} \cdot \overline{tv} \cdot \overline{tu}}$$

Výsledný logický výraz nyní realizujeme hradly NAND, přičemž každý negovaný součin v tomto výrazu představuje jedno hradlo NAND (tj. 6 dvouvstupových hradel a jedno šestivstupové hradlo). Logický obvod realizující zadanou funkci je vyobrazen na Obr. 34.

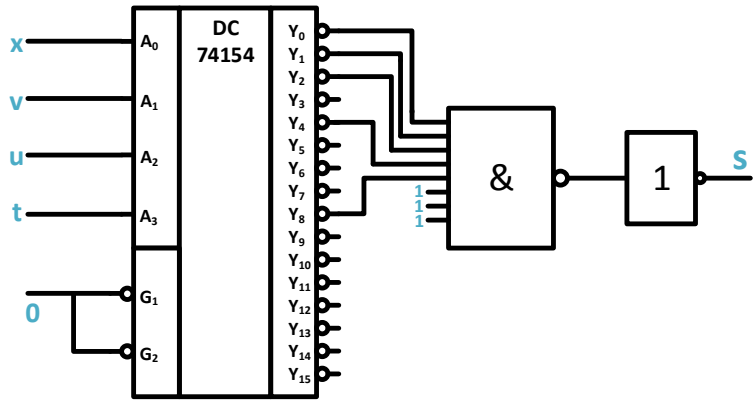
Použití dekodéru



Obr. 32: Realizace dané úlohy hradly NAND

Další možností jak realizovat danou úlohu, je použití dekodéru. V tomto případě postupujeme tak, že vybereme jednotlivé řádky tabulky s jednotkovou funkční hodnotou a logický součet těchto řádků bude tvořit výslednou funkci. Jelikož čísla řádků jsou volena tak, že odpovídají binárnímu zakódování jednotlivých proměnných, můžeme k sestavení obvodu použít dekodér. Dekodér totiž přiřazuje každému zakódovanému číslu řádku tabulky jeden výstup. V případě, že je na vstupu nastaveno nějaké číslo, bude aktivní příslušný výstupní vodič. Výstupní vodiče odpovídající řádkům s nenulovou funkční hodnotou můžeme logicky sečíst obvodem OR a získáme signál S, který aktivuje alarm.

Jelikož v příkladu použijeme dekodér 74151, který má výstupy negovány, musíme místo součtu výstupních vodičů pomocí obvodu OR provést negovaný součin obvodem NAND (viz De Morganova pravidla). Jelikož v pravdivostní tabulce je více než osm řádků s jedničkovou funkční hodnotou S, není možno realizovat výsledný negovaný součin pomocí jednoho hradla NAND. Proto jsme vytvořili



Obr. 33: Realizace dané úlohy s využitím dekodéru

obvod pro realizaci funkce S', kde S' je inversí funkce S. Funkci S jsme potom získali tím, že jsme funkční hodnotu S' invertovali pomocí invertoru. Toto zapojení můžeme vidět na následujícím obrázku (Obr. 35).

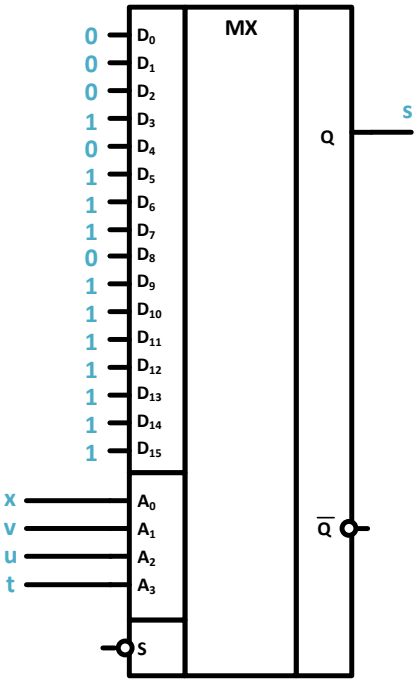
Použití multiplexoru

Třetí možností jak danou úlohu realizovat, je využití multiplexoru. Jelikož máme 4 vstupní proměnné (viz Tab. 11) můžeme pro realizaci využít 16ti vstupového multiplexoru, kde na adresové vstupy přivedeme naše vstupní proměnné (čidla T, U, V, X) a na datové vstupy multiplexoru přivedeme požadované výstupní hodnoty pro všechny kombinace vstupních proměnných. Realizace zadaného příkladu tímto způsobem je vyobrazena na Obr. 36. Tento způsob řešení je však dosti nehospodárný a proto pro realizaci použijeme 8mi vstupový multiplexor. Toto řešení je vyobrazeno na Obr. 37. V tomto případě použijeme jako adresu pouze tři ze vstupních proměnných a 4. vstupní proměnou (v přímém nebo negovaném tvaru) přivádíme spolu s log. 1 a log. 0 na datové vstupy multiplexoru v závislosti na vztahu mezi použitou vstupní proměnnou (v tomto případě proměnná X) a požadovanou hodnotou výstupy pro všechny kombinace zbylých vstupních proměnných (naznačeno v tabulce, která je součástí Obr. 37).

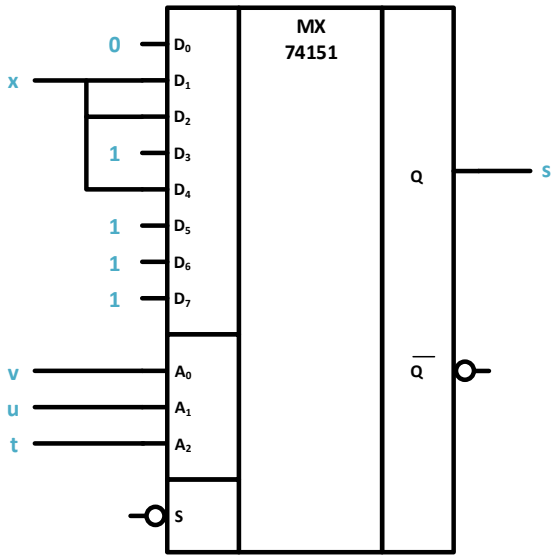
Tab. 22: Pravdivostní tabulka pro realizaci úlohy s využitím 16ti vstupového multiplexoru

Index	t	u	v	x	s
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

Tab. 23: Pravdivostní tabulka pro realizaci úlohy s využitím 8mi vstupového multiplexoru



Obr. 34: Realizace dané úlohy s využitím 16ti vstupového multiplexoru



Obr. 35: Realizace dané úlohy s využitím 8mi vstupového multiplexoru

Index	t	u	v	x	s	
0	0	0	0	0	0	0
1	0	0	0	1	0	
2	0	0	1	0	0	x
3	0	0	1	1	1	
4	0	1	0	0	0	x
5	0	1	0	1	1	
6	0	1	1	0	1	1
7	0	1	1	1	1	
8	1	0	0	0	0	x
9	1	0	0	1	1	
10	1	0	1	0	1	1
11	1	0	1	1	1	
12	1	1	0	0	1	1
13	1	1	0	1	1	
14	1	1	1	0	1	1
15	1	1	1	1	1	

5.4 Úlohy k procvičení látky

V následujících úlohách sestavte pravdivostní tabulku slovně popsaných logických funkcí, minimalizujte výraz pomocí Karnaughových map a realizujte zapojení obvodu pomocí hradel NAND. Alternativně vyřešte s pomocí dekodéru a multiplexoru.

Spouštěni elektromotoru

Před zapnutím trojfázového elektromotoru /nulové otáčky/, je nutné připojit kartáčky a zařadit

spouštěcí odpor. Po spuštění je nutno vyřadit spouštěcí odpor a odpojit kartáčky. Navrhněte logický obvod, který vyšle výstražný signál, jestliže v klidovém stavu nejsou zapojeny kartáčky a zařazen odpor nebo při běhu motoru jsou zapojeny kartáčky nebo zařazen odpor, nebo se po zapnutí motor nerozběhne.

Log. proměnné: Zapnutý motor, nenulové otáčky, zařazený odpor, zapnuté kartáčky.

Log. funkce: Výstražný signál, Ovládání světel automobilu

Světla automobilu

V automobilu je možno zapnout tato světla: parkovací, tlumená, dálková a mlhová. Platí tato pravidla: Při zapnutí tlumeného, dálkového nebo mlhového světla se musí automaticky rozsvítit světlo parkovací. Mlhová světla svítí pouze tehdy, když nesvítí světla dálková. Při současném požadavku na rozsvícení tlumených a dálkových světel se rozsvítí světla dálková. Navrhněte logický obvod, který pro libovolnou kombinaci tlačítek zajistí správné rozsvícení světel i se zřetelem na bezpečnost silničního provozu.

Log. proměnné: tlačítka parkovacích, tlumených, dálkových a mlhových světel.

Log. funkce: napětí na vodičích vedoucích k jednotlivým žárovkám parkovacích, tlumených, dálkových a mlhových světel.

Nápojový automat

Automat obsahuje otvor pro vhození malé či velké mince a dvě tlačítka pro volbu sodovky nebo limonády. Po vhození malé mince a stisknutí tlačítka sodovka automat nalije sodovku, po vhození velké mince a stisknutí tlačítka limonáda nalije limonádu. Při chybné volbě automat vrátí minci. Automat nesmí šidit ani majitele ani zákazníky.

Log. proměnné: vhozena velká mince, vhozena malá mince, zvolena sodovka, zvolena limonáda

Log. funkce: napětí na elektromagnetu pro nalití sodovky, limonády a pro vrácení mince.

Ochrana parního kotle

Parní kotel má čtyři hořáky, na každém z nich je čidlo, které signalizuje, zda plamen hoří či nehoří. Navrhněte logický obvod, který signalizuje poruchu, hoří-li tři nebo méně hořáků a uzavře přívod plynu, zhasnou-li dva nebo více hořáků.

Log. proměnné: Indikace hoření jednotlivých hořáků

Log. funkce: Poruchový signál, elektromagnet ovládající přívod plynu

---

Jednobitová sčítačka

Navrhněte logický obvod pro součet a přenos jednobitové sčítačky. Sčítají se dva jednobitové sčítanca s přihlédnutím na přenos z nižšího řádu, výsledkem je součet a přenos do řádu vyššího.

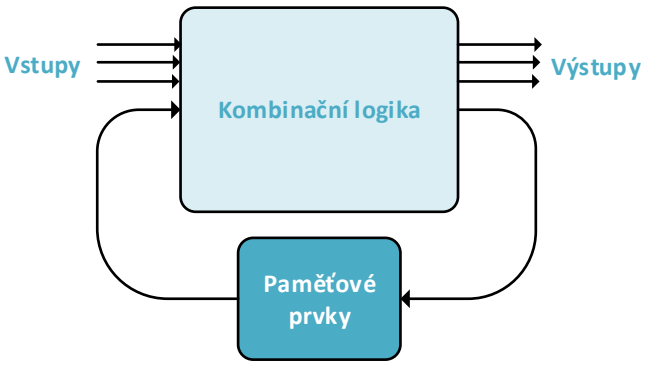
Log. proměnné: Dva sčítance, přenos z nižšího řádu

Log. funkce: Součet, přenos do vyššího řádu

---

6 SEKVENČNÍ OBVODY

Sekvenční obvod je takový obvod, jehož výstupy jsou určeny hodnotou vstupů a vnitřním stavem. Na Obr. 38 vidíme, že při konstrukci sekvenčních obvodů bychom měli dodržet pravidlo, které neumožňuje vkládat zpětné vazby do kombinačního obvodu, bez toho aniž by tyto vazby vedly přes paměťové členy, tj. obvody, které jsou schopny definovaným způsobem uchovávat informace. Tento přístup k sekvenčním obvodům usnadňuje návrh, neboť zjednodušuje analýzu chování obvodu. V této kapitole se budeme zabývat zvlášť jednotlivými typy paměťových prvků tj. klopnými obvody a pamětmi, základními sekvenčními automaty tj. čítači a ukážeme možnost jednoduchého návrhu obecných automatů.



Obr. 36: Obecné blokové schéma sekvenčního obvodu

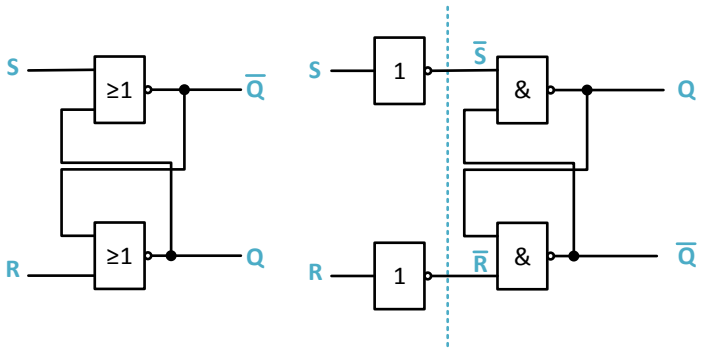
6.1 KLOPNÉ OBVODY

Klopný obvod (KO) je elektronický obvod, který se skokové překlápí mezi dvěma napěťovými stavy. Klopné obvody rozdělujeme na bistabilní, monostabilní a astabilní. Astabilní obvody slouží ke generování periodického signálu, který může být využit například pro synchronizaci. Monostabilní obvody upravují zachycené impulsy na impulsy s předem stanovenou délkou. Bistabilními KO, na které se zaměříme v následující kapitole, jsou KO se dvěma stabilními stavy (tj. pro změnu stavu je zapotřebí vnější podmět) a používají se jako paměťové prvky (anglicky flip-flops). Tyto KO dělíme na asynchronní a synchronní. Synchronní obvody reagují na vstupní signály pouze v okamžicích, kdy je aktivní hodinový signál, povolující změnu stavu. Asynchronní obvody reagují na všechny změny vstupního signálu. Asynchronní obvody jsou jednodušší než synchronní a proto se jimi budeme zabývat dříve.

6.1.1 RS KLOPNÝ OBVOD

Vstupní proměnné jsou R a S výstupní proměnná je Q a její negace . Hodnota Qt v pravdivostní tabulce (Tab. 12 a Tab. 13) odpovídá hodnotě výstupu Q v okamžiku před změnou jedné ze vstupních proměnných na hodnotu uvedenou v tabulce, Hodnota Qt+1 udává hodnotu výstupu Q po přivedení příslušných hodnot na vstupy R a S při původní hodnotě výstupu Qt.

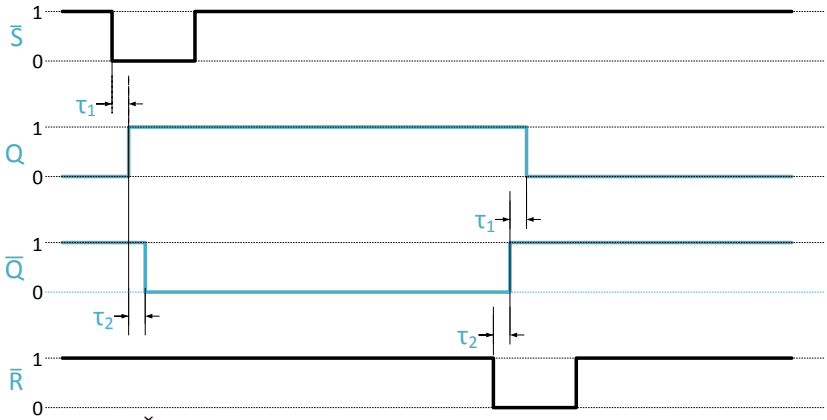
Obvod RS je paměťový prvek sloužící k zapamatování 1 bitu informace. Jestliže je aktivní vstup S (SET) zapisuje se do obvodu log. 1, je-li aktivní vstup R (RESET) zapisuje se do paměti log. 0. Není-li žádný ze vstupů aktivní, nemění se obsah paměti, pamatuje se předchozí stav. U obvodu tvořeného hradly NOR jsou aktivní úrovně signálů R a S logické 1. U obvodu tvořeného výhradně obvody NAND jsou aktivními úrovněmi signálů a úrovně log. 0. Proto se k tomuto obvodu mohou přidat vstupní invertory (viz Obr. 39), které změní aktivní úrovně tak, aby byly v souladu s běžnou konvencí. RS klopné obvody tedy v případě aktivního signálu R nulují výstup Q, v případě aktivního signálu S nastavují výstup a v případě obou vstupů neaktivních si pamatují předchozí stav (Tab. 12 a Tab. 13).



Obr. 37: RS klopný obvod tvořený hradly NOR a NAND

Je nutno zabránit tomu, aby byly současně aktivní oba vstupy, protože na výstupech se objeví zakázaný stav, kdy neplatí předpoklad o tom, že je inverzí signálu Q. Tohoto stavu je třeba se vyvarovat v zapojeních s obvody RS, neboť může vést k nedefinovanému následujícímu stavu.

Časový diagram RS klopného obvodu je na Obr. 40. Na diagramu je patrný vliv zpoždění signálu při průchodu jednotlivými hradly v okamžicích překlápění stavu klopného obvodu.



Obr. 38: Časový diagram RS obvodu tvořeného hradly NAND

Tab. 24: Pravdivostní tabulka RS KO tvořeného hradly NOR. Zakázané stavy jsou označeny x.

R	S	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0x
1	1	1	0x

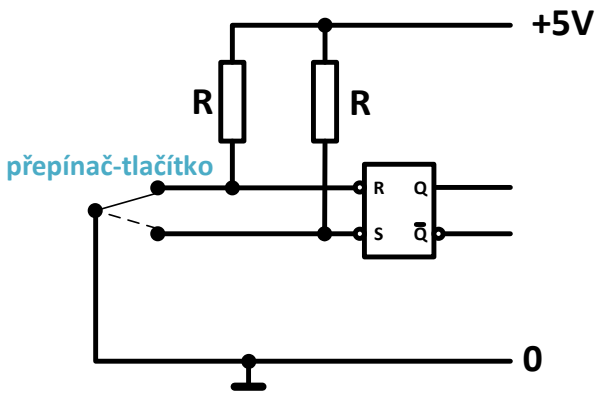
Tab. 25: Pravdivostní tabulka RS KO tvořeného hradly NAND Zakázané stavy jsou označeny x.

$\overline{R}$	$\overline{S}$	Q_t	Q_{t+1}
0	0	0	1x
0	0	1	1x
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Jako příklad využití RS klopného obvodu si nyní ukážeme úlohu na ošetření tlačítka proti zákmitu.

6.1.2 Ošetřené tlačítko

Mechanický přepínač – tlačítko při sepnutí nebo rozepnutí obvodu způsobí několik zákmitů výstupního napětí. Jelikož elektronické obvody na tyto nežádoucí zákmity reagují, je třeba kritická tlačítka ošetřit. Ošetřené tlačítko (přepínač) lze získat z obvodu RS s negovanými vstupy. Schéma zapojení je na Obr. 41.



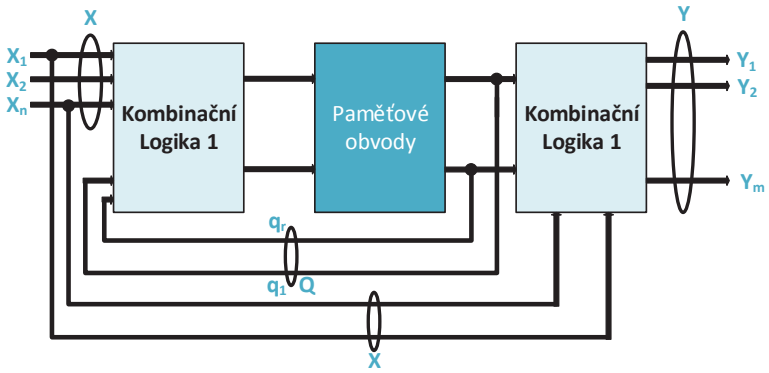
Obr. 39: Ošetřené tlačítko s RS KO

Odpor R volíme v řádu kΩ. V naznačené poloze přepínače je R = 0, S = 1, takže podle pravdivostní tabulky RS obvodu je Q = 0. Při přepínání, je-li kontakt v mezipoloze je R = 1, S = 1, takže se zachová předchozí stav Q = 0, při prvním styku s dolním kontaktem je R = 1, S = 0, obvod se překlopí a Q = 1. Odskočí-li kontakt, dojde k přerušení dotyku, nastaví se R = 1, S= 1 avšak stav Q = 1 zůstává zachován. Na výstupu Q tedy vznikne pouze jeden přechod z úrovně 0 na úroveň 1, tedy i při vícenásobném odskočení kontaktu dojde k vytvoření pouze jedné vzestupné hrany při stisknutí tlačítka. Analogicky lze analyzovat pohyb kontaktu směrem vzhůru, který vzniká při uvolnění tlačítka, kdy dojde k vytvoření pouze jedné sestupné hrany výstupního signálu.

6.2 Návrh asynchronního sekvenčního obvodu

Asynchronní sekvenční obvod obsahuje alespoň jednu zpětnou vazbu. Stav asynchronního obvodu tedy závisí nejen na hodnotách vstupních proměnných ale i na hodnotě minulého vnitřního stavu.

Obecně je návrh asynchronního obvodu poměrně komplikovaným úkolem, neboť při návrhu je třeba respektovat všechny možné hazardy, které mohou vzniknout v kombinační části obvodu a je třeba zajistit stabilitu jednotlivých stavů tak, aby se projevily na výstupu obvodu.



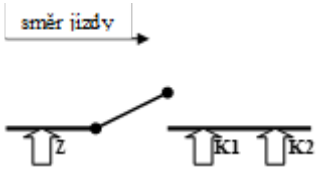
Obr. 40: Typy sekvenčních automatů - obecné schéma

Dále je třeba rozhodnout, jaký typ automatu chceme zkonstruovat, základní dva typy automatu jsou uvedeny na Obr. 42. Kde vektor X popisuje vstupní proměnné, vektor Y proměnné výstupní, vektor Q popisuje vnitřní stavy. KLO1 a KLO2 je kombinační logika, PO jsou paměťové (klopné) obvody. Jestliže vstupy X ovlivňují pouze kombinační logiku KLO1 a nikoliv KLO2, jedná se o automat typu Moore. V opačném případě hovoříme o automatu typu Mealy.

Tento obrázek popisuje nejen asynchronní sekvenční obvod ale i obvod synchronní. Aby úloha syntézy obvodu byla jednodušší, nepředpokládáme v sekvenčním obvodu existenci žádné zpětné vazby, která v sobě neobsahuje paměťový prvek. Tento předpoklad nám umožní využít k návrhu standardní klopné obvody a usnadní analýzu chování obvodu. Postup návrhu jednoduchého asynchronního sekvenčního obvodu budeme ilustrovat na příkladu.

6.2.1 Železniční přejezd

Na železničním přejezdu jezdí vlaky jedním směrem, zleva doprava podle Obr. 43. Přejede-li lokomotiva bod Z, závory se spustí, je-li poslední vagón za bodem K1, závory se zvednou. Sestavte obvod, který ovládá signál pro spuštění závor.



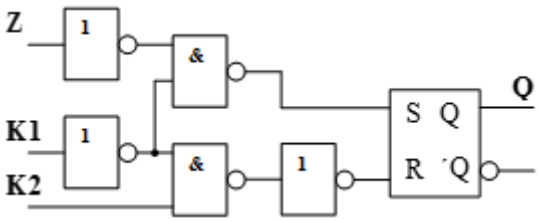
Obr. 41: Železniční přejezd. Z, K1 a K2 jsou kontakty, spínané vlakovou soupravou

Řešení je možné při použití třech spínačů v bodech Z, K1 a K2, které sepnou v případě přítomnosti některé části vlaku nad spínačem a pomocí obvodu, který vyhodnotí, ve kterém místě se nachází vlaková souprava. Akční veličinu, ovládající spouštění závor označíme jako Q. Je-li Q=1, závory se spustí. Jelikož pro správnou funkci závor postačuje jednobitová informace o tom, jestli se vlak nachází na místě, pro které je třeba spustit závory nebo ne, budeme tuto informaci uchovávat v jednom paměťovém prvku – v našem případě v RS klopném obvodu. Podmínky pro spuštění závor vyjádříme pravdivostní tabulkou:

Tab. 26: Podmínky pro spuštění závor

číslo řádku	Z	K1	K2	Q_t	Q_{t+1}	funkce RS obvodu
1	0	0	0	0	0	pamatování
2	0	0	0	1	1	pamatování
3	0	0	1	x	0	nulování
4	0	1	x	x	1	nastavení
5	1	x	x	x	1	nastavení

V souladu s předpoklady jsme pro výsledný signál Qt+1 určili, že v případě, když žádný z kontaktů nehlásí přítomnost vlaku, bude zapamatován předchozí stav, v případě, že alespoň jeden z kontaktů Z nebo K hlásí přítomnost vlaku, budou závory spuštěny nezávisle na předchozím stavu, a v případě, že K2 hlásí přítomnost vlaku a současně ostatní kontakty hlásí nepřítomnost vlaku, závory budou zvednuty. Jestliže porovnáme pravdivostní tabulku úlohy s pravdivostní tabulkou RS klopného obvodu (Tab. 12 a Tab. 13) vidíme, že RS klopný obvod můžeme ovládat tak, aby na jeho výstupu byla logická hodnota odpovídající signálu pro ovládání závor. Budící funkce RS obvodu potom budou realizovány následovně: na vstup S přivedeme hodnotu log. 1 v případě, že Z nebo K1 jsou rovny 1. (De Morganovými pravidly upravíme výraz S = Z +K1 =). Při této hodnotě vstupu S budou závory spuštěny. Na vstup R přivedeme hodnotu log. 1 v případě, že platí současně, že K1 = 0 a K2 = 1. Při této hodnotě vstupu R budou závory vytaženy. Případ, kdy současně oba vstupy S i R jsou rovny jedné, nemůže nastat. Jestliže jsou všechny vstupní signály nulové, na S i R vstupy přivádíme 0. Při této hodnotě vstupů setrvávají závory v poloze dané předchozím nastavením. Výsledné schéma zapojení obvodu je na Obr. 44.



Obr. 42: Schéma obvodu pro ovládání závor

6.3 ÚLOHY NÁVHRU SEKVENČNÍCH OBVODŮ

Teplota v peci

Teplota v peci je snímána čidlem připojeným na analogově digitální převodník. Z jeho výstupu jsou odvozeny dvě logické proměnné H a D. Proměnná H nabývá log. 1, stoupne-li teplota nad stanovenou mez TH, proměnná D je v log.1 při poklesu teploty pod povolenou hodnotu TD. Navrhněte obvod, který zapne topení, klesne-li teplota pod TD a vypne jej při překročení teploty TH.

Log. proměnné: Teplota pod TD, teplota nad TH

Log. funkce: Ovládání topení, Automatické startování automobilu

Elektromotor 1

Sestavte logický obvod, který vypne elektromagnet startéru, jestliže motor naskočil a nedovolí startovat při běžícím motoru. Startuje se stiskem tlačítka a startér běží tak dlouho, dokud motor nedosáhne předepsaných otáček, signalizovaných log.1 příslušného čidla. Výstupní logická proměnná ovládá elektromagnet startéru.

Log. proměnné: Tlačítko startéru, indikace běžícího motoru

Log. funkce: Ovládání elektromagnetu startéru, Spouštění elektromotoru

Elektromotor 2

Elektromotor se spouští tlačítkem Z a zastavuje tlačítkem V. Navrhněte logický obvod, který zajistí, že elektromotor běží po stisku tlačítka Z a zastaví se po stisku tlačítka V i v případě, že je současně stisknuto tlačítko Z. Výstupní logická proměnná ovládá stykač, hodnota log.1 = zapnuto.

Log. proměnné: Tlačítko <Z>apni, <V>ypni

Log. funkce: Ovládání stykače motoru, Regulace výšky hladiny

Hladina nádrže

Stav hladiny v nádrži je sledován třemi snímači A, B, C. Snímač A je nejvýše, Snímač C nejníže. Na výstupu snímače je log.1, jestliže není ponořen. Voda je čerpána dvěma čerpadly - hlavním a vedlejším. Klesne-li hladina pod B, zapne se hlavní čerpadlo; klesne-li pod C zapne se i pomocné čerpadlo. Obě čerpadla se vypnou, stoupne-li hladina nad snímač A. Sestavte logický obvod, který zajistí automatické zapínání a vypínání čerpadel.

Log. proměnné: Čidlo úrovně hladiny A, B, C

Log. funkce: Ovládání Hlavního a Pomocného čerpadla, Automatické závory

Zabezpečení železniční tratě

Na trati jezdí vlaky jen jedním směrem. Před závorami je kontakt S, za nimi v dostatečné vzdálenosti kontakt Z. Kontakt je sepnut, je-li nad ním vlak a v tom případě je na jeho výstupu log.1. Navrhněte logický obvod, který spustí závory, najede-li lokomotiva nad kontakt S a zvedne je, jakmile vlak najede na kontakt Z.

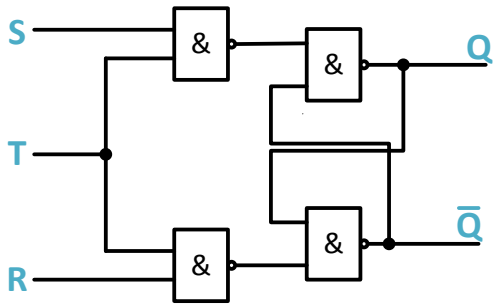
Log. proměnné: Nájezdový kontakt Z a S.

Log. Funkce: Ovládání závor

6.4 Synchronní klopné obvody

Asynchronní klopné obvody reagují okamžitě po změně vstupních proměnných, což může vést k obtížím při návrhu sekvenčních obvodů, neboť obvody připojené na vstupu klopného obvodu mohou vlivem hazardů způsobit nechtěné překlopení obvodu. Při návrhu asynchronních obvodů je tedy třeba důsledně všechny možné hazardy odstranit a zajistit stabilitu jednotlivých stavů. Takovýto způsob návrhu je obtížný, pro složitější obvody je prakticky neřešitelný a proto se více než asynchronních obvodů v praxi využívá obvodů synchronních, které reagují na vstupní signály pouze v těch okamžicích, kdy jsou všechny logické hodnoty na vstupu ustáleny. Výstupní hodnota synchronních obvodů je tedy určena přesně bez ohledu na možné hazardy v předcházejících obvodech.

6.4.1 RS klopný obvod



Obr. 43: Synchronní RS klopný obvod

Na Obr. 45 je upraven RS klopný obvod tak, aby pracoval jako synchronní. Obvod reaguje na vstupy R a S pouze tehdy, když vstup T je nastaven do log. 1. Vidíme, že synchronní obvody mají navíc oproti asynchronním obvodům další synchronizační vstup, který umožňuje znecitlivět ostatní vstupy až do doby, kdy jsou zajištěny podmínky pro správnou funkci obvodu (předcházení

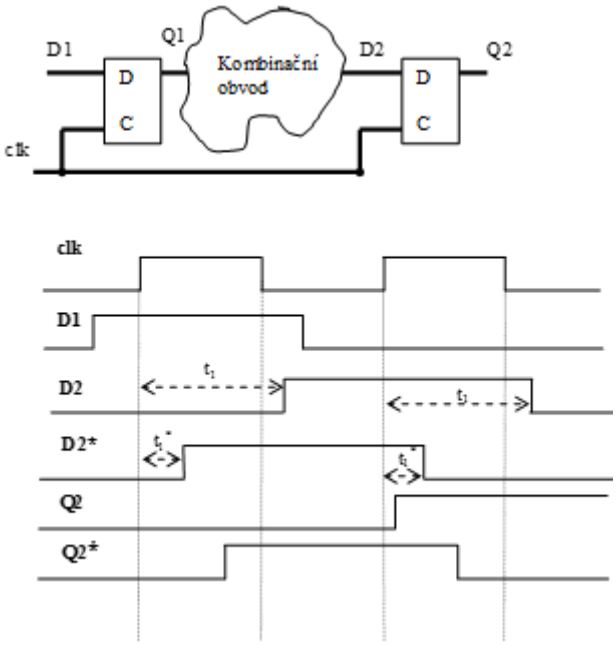
hazardům). Tento vstup bývá označen zpravidla zkratkou T, C nebo CLK, slovně jej popisujeme jako hodinový vstup. Hodinové vstupy bývají buzeny zpravidla periodickým signálem, tvořeným krátkými impulsy a nazývaným hodinový signál.

Další vylepšení synchronního RS klopného obvodu je uvedeno na Obr. 46. Aby bylo zabráněno vzniku zakázaného stavu, je signál R vytvořen ze signálu S (D) pomocí invertoru. Je vidět, že vstupní hodnoty 11 a 00 RS klopného obvodu nemohou nastat, obvod reaguje na hodnotu 0 i 1 na vstupu D a v případě , že na vstupu C je nastavena log.1, zapíše vstupní hodnotu z D na výstup. Pravdivostní tabulka jednostupňového D klopného obvodu je v Tab. 16.

Tab. 27: Pravdivostní tabulka jednostupňového D KO

D	C	Q_{t+1}
0	0	Q_t
1	0	Q_t
0	1	0
1	1	1

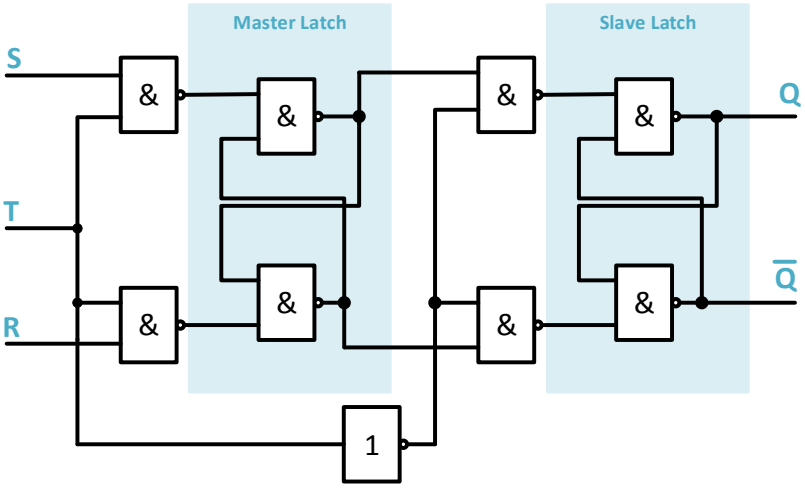
Jednostupňový D klopný obvod je základním prvkem pro konstrukci statických pamětí. U těchto pamětí je k D klopným obvodům připojena dekódovací logika, která umožňuje adresaci. Jestliže je klopný obvod adresován a přijde povolení zápisu (signál C), pak se do obvodu zapíše hodnota na vstupu D, jestliže povolení zápisu nepřijde, pak paměť pracuje v režimu čtení a hodnota uložená v klopném obvodu je přepnuta na výstup z paměti. Synchronní obvody, které jsme doposud popsali, patří k těm nejjednodušším. Tyto obvody není možno využít při návrhu složitějších sekvenčních schémat, neboť zde dochází k hazardním stavům, jak je dokumentováno na příkladu zobrazeném na Obr. 47. Výstupní hodnota Q2 je ovlivněna rychlostí šíření signálu v kombinační logice a uvnitř prvního klopného obvodu. Rozdílné zpoždění signálu (znázorněno jako t1 a t1* způsobí, že průběh vstupního signálu D2 může být změněn na D2* a tudíž i časový průběh výstupního signálu Q2 se změní na Q2*.



Obr. 44: Problém hazardu vznikajícího ve vícestupňovém sekvenčním obvodu.

Při návrhu se většinou využívají dvoustupňové (Master-Slave) obvody (Obr. 48). Tyto obvody zaručují kromě definovaného okamžiku zápisu do klopného obvodu i definovaný okamžik rozšíření vnitřního stavu na výstup obvodu. Tato vlastnost je nezbytná pro konstrukci většiny sekvenčních obvodů, neboť umožňuje konstruovat zařízení tak, že v daný okamžik reagují pouze

určené klopné obvody. Reakce synchronního Master-Slave obvodu po příchodu hodinového impulsu sestává ze dvou na sebe navazujících fází: zapamatování informace ze vstupu v prvním klopném obvodu a zpracování informace a její přenos na výstup v obvodu druhém.



Obr. 45: Master-slave RS klopný obvod

Na Obr. 48 je vyobrazeno ideové schéma master-slave RS klopného obvodu. První stupeň obvodu je řízen hodinovým signálem T, aktivním v log. 1, druhý stupeň je řízen invertovaným hodinovým signálem T.

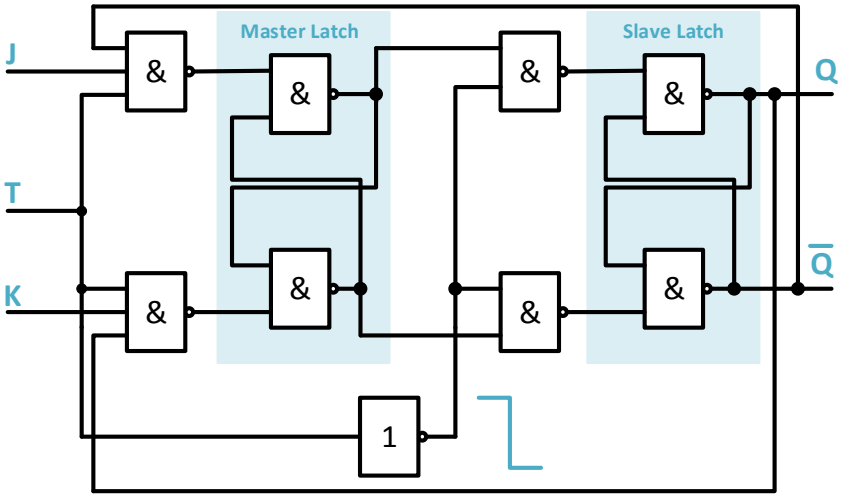
Počátek fáze zapamatování je vždy řízen hodinovým impulsem. Fáze přenosu může buď následovat bezprostředně po fázi zapamatování, nebo může být řízena hodinovým impulsem. Hodinové impulsy jsou buď přivedeny dvěma zvláštními vodiči ze zdroje hodinových signálů, nebo se uvnitř obvodu odvodí z náběžné a sestupné hrany jednoho vnějšího hodinového signálu. Používají se čtyři způsoby řízení synchronního sekvenčního obvodu, které jsou popsány v Tab. 17.

Tab. 28: Časové diagramy reakcí na hodinový signál

řízení	hladinou	vzest. hranou	sest. hranou	dvěma hranami
hodin. signál	t_s t_h	t_s t_h	t_s t_h	t_s t_h
zapam. a přenos				
hod. vstup	C	C	C	C
pravd. tab.				

6.4.2 JK KLOPNÝ OBVOD

Principiální schéma JK klopného obvodu můžeme odvodit ze schématu Master-slave RS klopného obvodu, jak je to ukázáno na Obr. 48. Přidáním zpětné vazby z výstupů Q a Q/ na vstupní členy NAND (Obr. 49) je možno změnit chování obvodu na JK klopný obvod. Vstup J odpovídá vstupu S, K odpovídá vstupu R.



Obr. 46: Obr_49_JK_master_slave.pdf

JK klopný obvod nemá žádný zakázaný stav, jestliže na oba vstupy J i K přivedeme v okamžiku aktivní úrovně hodinového signálu aktivní úroveň (log. 1) obvod se překlopí do opačného stavu, než byl před příchodem hodinového pulsu. Synchronní obvody jsou většinou konstruovány tak, že umožňují asynchronní nastavení do stavu log. 1 na výstupu (signál S) a asynchronní nulování (signál R). Pravdivostní tabulka JK klopného obvodu je uvedena v Tab. 18. Obvod JK je vhodný zvláště pro konstrukci čítačů vzhledem ke své schopnosti invertovat stav při příchodu logických hodnot 11 na vstupy JK.

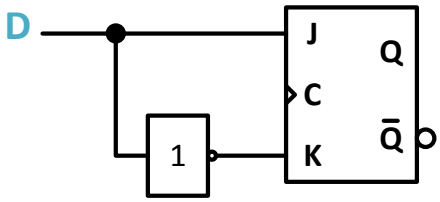
Tab. 29: Pravdivostní tabulka JK klopného obvodu s asynchronním nastavením a nulováním, hladinový typ

	Vstupy					Výstupy		Poznámky
	S	R	C	J	K	Q^+	$\bar{Q}^+$	
Asynchr.	0	1	x	x	x	1	0	nastavení do 1
	1	0	x	x	x	0	1	nastavení do 0
	0	0	x	x	x	1*	1*	nestabilní stav
Synchr.	1	1	⎍	0	0	Q^-	$\bar{Q}^-$	beze změny
	1	1	⎍	1	0	1	0	nastavení do 1
	1	1	⎍	0	1	0	1	nastavení do 0
	1	1	⎍	1	1	$\bar{Q}^-$	Q^-	změna stavu
	1	1	0	x	x	Q^-	$\bar{Q}^-$	beze změny

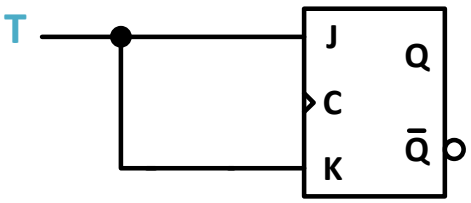
* Nestabilní stavy (obdobně jako u dvoustupňového klopného obvodu RS)

6.4.3 D a T klopné obvody

D klopný obvod vznikne propojením a invertováním jednoho ze vstupů (vstup K) u JK klopného obvodu (Obr 50). Pravdivostní tabulka Master-slave D klopného obvodu je shodná s pravdivostní tabulkou jednostupňového D klopného obvodu Tab. 15. D klopný obvod se používá zvláště pro konstrukci registrů a posuvných registrů.



Obr. 47: D klopný obvod



Obr. 48: JK klopný obvod

T klopný obvod vznikne z JK klopného obvodu spojením jeho vstupů (bez invertování). Úprava JK obvodu je naznačena na Obr. 51 a příslušná pravdivostní tabulka je v Tab 19. Použití T klopného obvodu je časté zvláště při konstrukci čítačů.

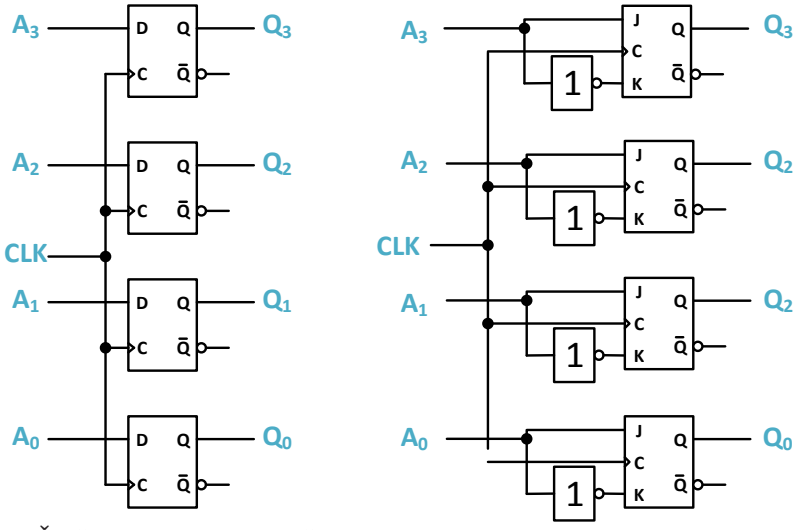
Tab. 30: Pravdivostní tabulka T klopného obvodu

T	Qt	Qt+1
0	0	0
0	1	1
1	0	1
1	1	0

7 VYŠŠÍ KONSTRUKČNÍ CELKY S KLOPNÝMI OBVODY

7.1 REGISTRY

N bitový registr je logický obvod s hodinovým vstupem, n informačními vstupy a s n výstupy. Může též obsahovat nulovací vstup popřípadě nastavovací vstup. Hodinový impuls zajistí přenos okamžitých hodnot z informačních vstupů na výstup. Není-li přítomen, obsah výstupu se nemění. Registr lze realizovat jak členy D, tak členy JK. Principiální schéma čtyřbitového registru z obvodů D je na Obr 52.



Obr. 49: Čtyřbitový registr tvořený hranovými klopnými obvody typu D a JK

Registr se mj. využívá k přenosu informace mezi dvěma kombinačními obvody. Registr zajistí, že po příchodu dostatečně zpožděného hodinového impulsu se zapamatují ustálené hodnoty výstupů předchozích částí obvodu.

POSUVNÉ REGISTRY

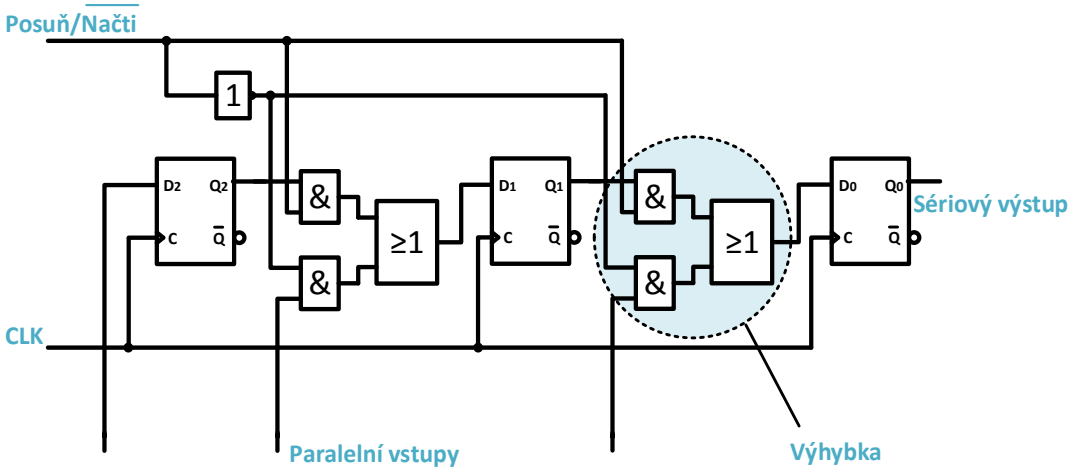
Posuvný n bitový registr obsahuje hodinový vstup C, jeden informační vstup a jeden nebo dva informační výstupy (Obr. 53). Zpravidla bývá navržen tak, aby jej bylo možno jedním signálem nulovat. Posuvný registr obsahuje zpravidla D klopné obvody hranového typu a jeho funkce je následující: Po příchodu hodinového impulsu se obsah vstupu Di přenese na výstup Qi pro všechna i = 0, 1, …, n. Tato funkce současně posune informace o jedno místo vpravo nebo vlevo v závislosti na směru šíření signálu.

Pomocí posuvných registrů lze snadno binárně dělit a násobit. Posun obsahu o jedno místo vpravo je podíl po dělení původního obsahu dvěma (se ztrátou zbytku po dělení), posun o jedno místo vlevo je výsledkem násobení původního obsahu dvěma. Posuvných registrů je více typů, liší se v počtu vstupů a výstupů. Běžně se využívá posuvných registrů se sériovým vstupem a výstupem a současně s paralelními vstupy nebo výstupy. Takovýto registr má dvě funkce, mezi kterými se přepíná pomocí zvláštního řídícího vstupu.

V základním režimu slouží jako běžný paralelní registr, ve druhém režimu slouží k sériovému zapsání popřípadě přečtení obsahu registru pomocí jednoho vodiče. Využití takovéhoto registru je tam, kde je třeba paralelní informace přenést pomocí jednoho vodiče. Příklad zapojení je na Obr. 54. Vstup pro řízení funkce je nazván POSUŇ/(NAČTI)‾. Paralelní data je možno zapsat jestliže je tento signál roven log. 0. Sériová data se zapisují prostřednictvím vstupu D3, sériové

čtení dat je umožněno prostřednictvím výstupu Q0.

Pomocí posuvného registru s vhodně volenými zpětnými vazbami je možno vytvořit zvláštní čítače, které mají minimální hardwarové nároky na kombinační logiku mezi jednotlivými klopnými obvody a které jsou výhodné tam, kde vyžadujeme vysokou frekvenci hodinových pulsů. Specializované posuvné registry umožňují posun informace nalevo i napravo a využívají se ve výpočetní technice.



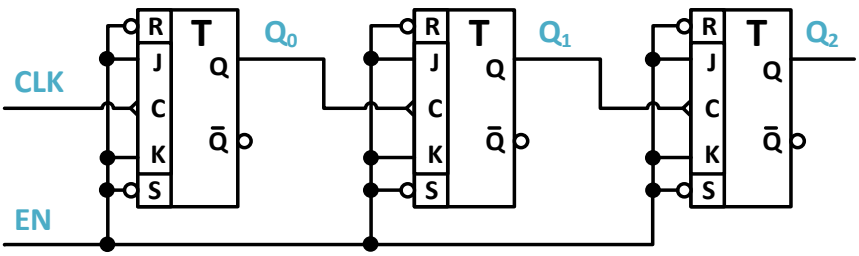
Obr. 50: Posuvný registr s paralelním zápisem

7.2 Čítače

Zjednodušeně řečeno, čítač je logický obvod, který zjišťuje počet došlých hodinových impulsů. Každý hodinový impuls změní jeho obsah, nejčastěji jej zvýší (sníží) o jednotku. Čítač však může čítat i jiným způsobem než v přirozeném pořadí binárního kódu (Grayův kód, binárně dekadický kód, další lineární i nelineární kódy). Čítače můžeme dělit podle počtu bitů, podle maximální frekvence hodinového signálu, podle typu kódu, který je čítačem generován, podle možnosti změny směru čítání nahoru nebo dolů nebo podle možností nastavení různých předvoleb.

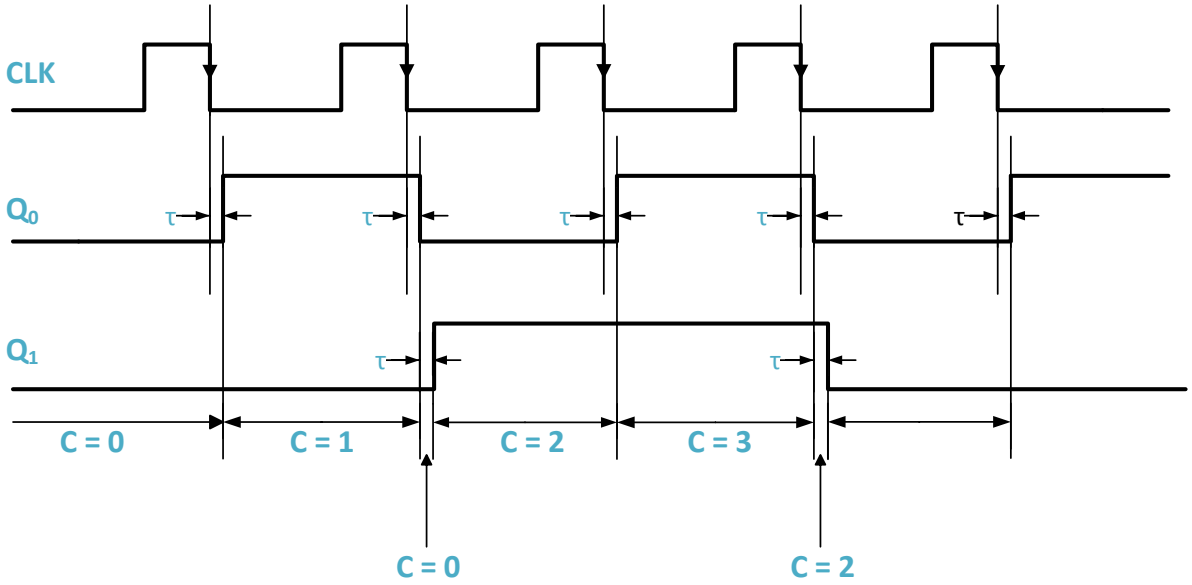
Čítače se využívají jako základní konstrukční prvek pro sekvenční automaty různých typů. Tvoří například jádro řadiče procesoru (viz dále), využívají se pro měření kmitočtu a periody, vytváření časových základen osciloskopů, řízení multiplexorů, k dělení frekvence různých signálů atd. Čítače mohou být asynchronní nebo synchronní. Toto označení neznamená, že asynchronní čítače nemají hodinový (synchronizační) vstup, ale že jednotlivé stupně asynchronního čítače jsou synchronizovány jiným než hodinovým signálem. Principiální schéma asynchronního čítače je na Obr. 55. Vzhledem k nastavení vstupů J, K a R jednotlivé klopné obvody mají funkci děliče hodinové frekvence dvěma.

7.2.1 Asynchronní čítač



Obr. 51: Tříbitový asynchronní čítač tvořený JK klopnými obvody. Q_0 tvoří nejnižší řád.

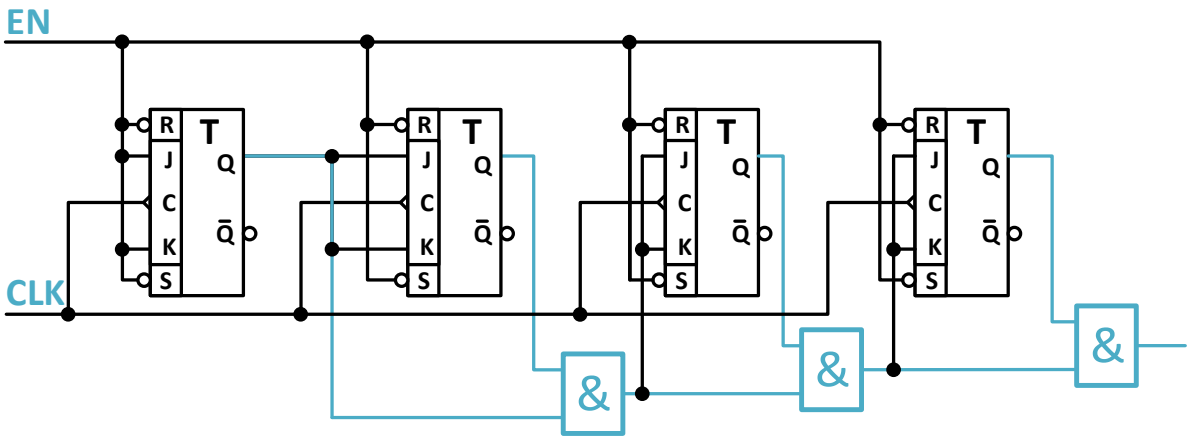
Každý obvod překlopí do opačného stavu při příchodu sestupné hrany hodinového impulsu. Jelikož hodinové impulsy pro vyšší řády jsou odvozeny vždy ze stavu řádu nižšího, dochází na klopných obvodech k vytváření posloupnosti binárního kódu, přičemž nejnižší řád je nalevo. Asynchronní čítač je jednoduchý, není třeba konstruovat zvláštní obvody, které generují přenos z nižších do vyšších řádů. Nevýhodou tohoto zapojení je ta skutečnost, že jednotlivé bity čítače jsou nastavovány do správné kódové hodnoty v nestejných okamžicích. Tuto skutečnost dokumentuje časový diagram na Obr. 56.



Obr. 52: Časový diagram asynchronního čítače. Na diagramu je vyznačeno zpoždění signálu τ při průchody jednotlivými stupni čítače

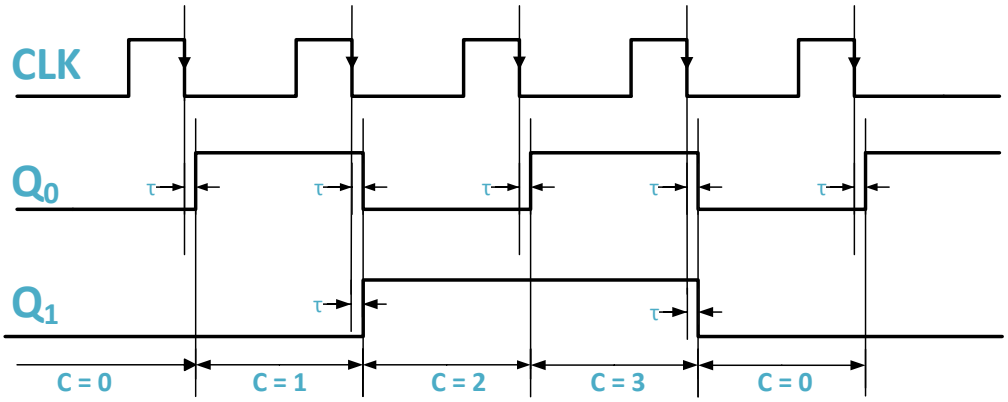
Na diagramu vidíme, že okamžik nastavení výstupu Q_0 je zpožděn oproti sestupné hraně hodinového impulsu o dobu τ, výstup Q_1 je zpožděn o dobu 2τ atd. Při větší délce čítače může dojít k tomu, že doba šíření signálu do nejvyššího řádu je delší než je perioda hodinového signálu a čítač tudíž nebude zobrazovat správné hodnoty generovaného kódu.

7.2.2 Synchronní čítač



Obr. 53: Synchronní čtyřbitový čítač tvořený T klopnými obvody

Principiální schéma čtyřbitového synchronního čítače je na Obr. 57. Z obrázku je patrné, že hodnota vyšších bitů čítače se změní při příchodu hodinového impulsu pouze tehdy, když v předchozích řádech byl po minulém hodinovém impulsu nastaven stav tvořený samými jedničkami.

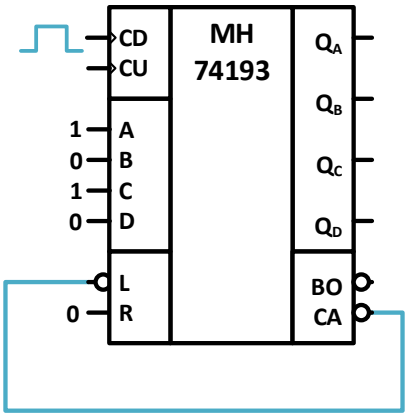


Obr. 54: Časový diagram dvou bitů synchronního čítače. Na diagramu je vyznačeno zpoždění signálu τ při průchody jednotlivými stupni čítače.

Výhodou synchronního čítače je to, že všechny bity čítače jsou nastavovány ve stejný okamžik. Přenos do vyššího řádu je nastavován kombinačními obvody, jejichž výstup se ustálí po nastavení nějakého stavu ještě před příchodem dalšího hodinového impulsu. Časový diagram je znázorněn na Obr. 58.

7.2.3 Zkrácení cyklu čítače

Úplným cyklem čítače rozumíme vzestupné nebo sestupné čítání v celém rozsahu čítače (např. 0 až 15 u čítače 74193). Neúplný cyklus je čítání v rozsahu N_1, N_2, kde $N_1 \geq 0$ a $N_2 \leq 15$. Např. pro $N_1 = 5$, $N_2 = 7$ bude na výstupu posloupnost čísel 5, 6, 7, 5, 6, ...



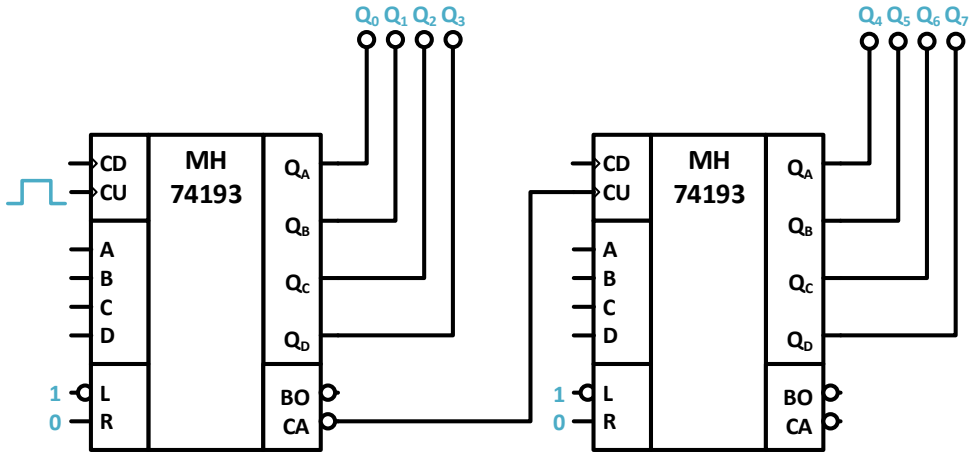
Obr. 55: Zkrácení cyklu čítače

Snadno lze realizovat neúplný cyklus pro čítání vpřed od N do 14 podle Obr. 59. Číslo N nazýváme předvolba. Předvolbu nastavíme na informačních vstupech A až D, nejnižší řád na svorce A. Na obr. B 7.5 je N = 5. Je-li na výstupu číslo 14, nastaví vzestupná hrana následujícího hodinového impulsu na výstupu číslo 15, ale sestupná hrana aktivizuje vstup L, výstup se přepíše číslem N a vzápětí přestane být výstup CA a tím i vstup L aktivním. Během hodinového impulsu je na výstupu číslo 15, ale po jeho ukončení je na výstupu číslo N.

7.2.4 Rozšířeni rozsahu čítače

Obvykle je rozsah jednoho čítače malý, pro zvětšení rozsahu se přidají další čítače, tím získáme vícestupňový čítač. V praxi se často používá dekadických čítačů 74192, pak každý stupeň odpovídá jednomu desítkovému řádu. Zapojení dvoustupňového čítače pro čítání vpřed je na Obr. 60. Hodiny se přivádějí na vstup prvního čítače. Jde-li o dekadický čítač, jsou na vstupu prvního čítače jednotky a na výstupu druhého desítky. Spojením vstupů L je možnost předvolby

na informačních vstupech obou čítačů.

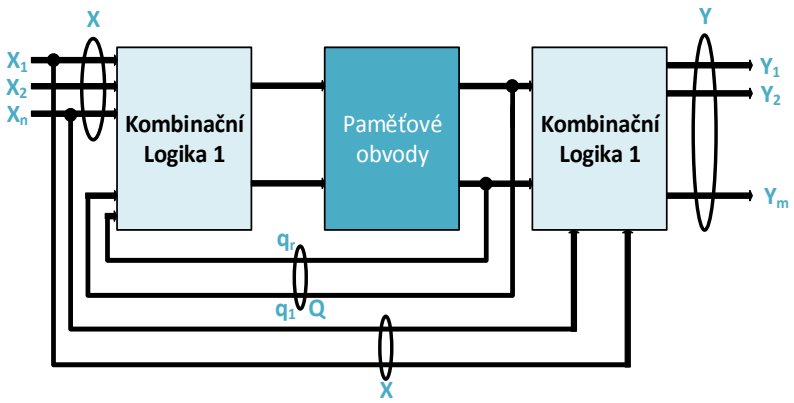


Obr. 56: Rozšíření cyklu čítače

8 NÁVRH SYNCHRONNÍCH SEKVENČNÍCH OBVODŮ

8.1 SEKVENČNÍ AUTOMATY

Obecně všechny systémy, kde počítáme s časem, jakožto s faktorem podmiňujícím chování jsou sekvenční. V praxi tuto sekvenčnost chování zanedbáváme a hovoříme například o kombinačních obvodech, kde při jejich zjednodušeném popisu nehovoříme o zpoždění průchodu signálu obvodem. Časovou podmíněnost výstupu však nelze zanedbat u obvodů, kde výstupy jsou podmíněny okamžitým vnitřním stavem. Sekvenční obvod (systém) je určen množinou vstupních symbolů (X) tzv. vstupní abecedou, množinou výstupních symbolů (Y) tzv. výstupní abecedou, množinou vnitřních stavů (Q), přechodovou funkcí (δ) a výstupní funkcí (λ). Přechodová funkce určuje, za jakých podmínek přechází automat mezi jednotlivými stavy a výstupní funkce určuje, jaké výstupní symboly jsou přiřazeny jednotlivým stavům a vstupům. Jestliže požadujeme, aby automat byl na začátku své funkce v definovaném stavu (Q0) je informace o tomto stavu také vyžadována pro popis chování automatu. Z kombinačního obvodu vytvoříme obvod sekvenční tak, že do obvodu zařadíme zpětnou vazbu, která hodnoty z výstupů nebo hodnoty některých vnitřních proměnných přivede k dalšímu zpracování některým kombinačním podobvodem, který je ve schématu obvodu blíže ke vstupům. Takto vytvořený systém může trpět tím, že zpracování signálů v různých částech kombinační sítě netrvá stejnou dobu a tudíž mohou vznikat různé hazardní jevy, které se obtížně předvídají a odstraňují. Z těchto důvodů jsou zpětné vazby v systému doplněny o synchronní klopné obvody, které zabezpečí, že systém reaguje na vstupy vždy bezhazardně. Obecné schéma automatu je na Obr. 61.



Obr. 57: Obecné schéma sekvenčního automatu

Při využití popisu automatu pomocí uspořádané pětice <X, Y, Q, δ, λ> můžeme rozlišit různé typy automatů.

Jestliže některá funkce nebo některé proměnná chybí, hovoříme o speciálních automatech:

Automat, který je definován čtveřicí < Y, Q, Γ, Φ> bude automatem autonomním. Za tento automat můžeme považovat například zdroj hodinových impulsů. Takový automat po zapnutí napájecího zdroje pracuje bez vstupních řídících signálů.

Automatem bez vnitřních stavů by byl kombinační obvod, a proto se jím zde nebudeme zabývat.

Automat bez výstupu se nazývá automatem Medveděvovým. Takový automat nemá praktické využití, protože jeho chování je z vnějšku nepozorovatelné.

Automat, který má všechny proměnné a funkce může být buď typu Moore nebo Mealy. Přehled typů automatů nalezneme v Tab. 19.

Tab. 31: Typy sekvenčních automatů

	Typ automatu	Určení automatu	Poznámka
1	Automat kombinačního typu – kombinační logický systém	$A = \{X, Y, \delta=F\}$	Automat má jeden konstantní stav (vnitřní).
2	Medvěděvův "konečný" automat – automat bez výstupního zařazení	$A = \{X, Q, \delta\}$ $Q^t = \delta(X^t, Q^{t-1})$	Výstupní chování automatu nelze pozorovat.
3	Mooreův automat	$A = \{X, Y, Q, \delta, \lambda_0\}$ $Q^t = \delta(X^t, Q^{t-1})$ $Y^t = \lambda_0(Q^t)$	Výstupní stav je závislý na vnitřním stavu automatu. Převoditelný na Mealyho.
4	Autonomní automat	$A = \{Y, Q, \delta, \lambda\}$ $Q^t = \delta_0(Q^{t-1})$ $Y^t = \lambda_0(Q^t)$	Konstantní chování z hlediska "vstupů".
5	Mealyho automat	$A = \{X, Y, Q, \delta, \lambda\}$ $Q^t=\delta(X^t, Q^{t-1})$ $Y^t=\lambda(X^t, Q^{t-1})$	Nejobecnější automat, výstupní stav je závislý na vnitřním stavu a na vstupu, převoditelný na Moora.
6	Stochastický, pravděpodobnostní automat	$A = \{X, Y, Q, \delta, \lambda\}$ $\delta: P_\delta \{Q_j / Q_i, X_1\}$ $\lambda: P_\lambda \{Y_m / Q_i, X_1\}$	Přechodové a výstupní funkce jsou pravděpodobnostmi P_δa P_λ. Realizace je deterministická.

Chování automatu typu Moore je dáno následujícími rovnicemi:

$Q^t = \delta(X^t, Q^{t-1})$ přechodová funkce

$Y^t = \lambda(Q^t)$ výstupní funkce

Uvedené rovnice říkají, že stav v časovém okamžiku t je určen vstupem v tomtéž čase a předchozím stavem (čas t-1), výstup v čase t je funkcí současného stavu. Jelikož funkce λ je kombinační funkcí, platí, že pro různé výstupy musí existovat různé vnitřní stavy.

Chování automatu typu Mealy je dáno následujícími rovnicemi:

$Q^t = \delta(X^t, Q^{t-1})$ přechodová funkce

$Yt = \lambda(X^t, Q^{t-1})$ výstupní funkce

Automat typu Mealy je tedy podobný automatu Moorovu s tím rozdílem, že výstup je funkcí aktuálního vstupu a stavu v předchozím taktu. Různým výstupům tedy nemusí odpovídat různé vnitřní stavy.

Z praktického hlediska existují dva základní rozdíly mezi oběma automaty:

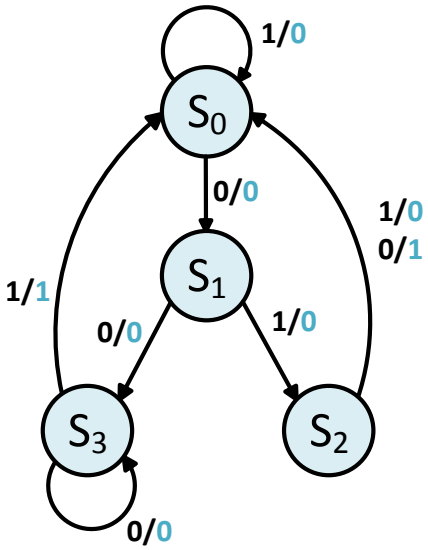
- Automat typu Mealy je zpravidla jednodušší než automat Mooreův, protože pro vytvoření požadované výstupní sekvence není třeba navrhovat pro každou hodnotu požadovaného výstupu, který má být odezvou na vstupní sekvenci, odpovídající vnitřní stav.
- Automat typu Mealy reaguje na vstupní sekvenci dříve než Automat typu Moore. Tato vlastnost může být nevýhodou, protože okamžik, kdy automat zareaguje na změnu vstupu, není

přesně definován, záleží na vnějším vstupním signálu, který nemusí být přesně synchronizován s hodinami automatu.

8.2 Popis chování automatů

Sekvenční automaty lze zadávat několika způsoby. Nejčastěji se popisují pomocí grafu přechodů a tabulky přechodů a výstupů. Graf přechodů umožňuje snazší pochopení funkce navrhovaného automatu, a proto je vhodné jej využít ve fázi návrhu chování automatu. Tabulka je základem pro další syntézní kroky a proto je vhodné ji využít v těch případech, kdy manuálně provádíme celou syntézu. Vlastní syntéza automatu je poměrně složitou úlohou zvláště pro automaty, které mají velký počet vnitřních stavů. Některé automatizované návrhové systémy však tuto úlohu řeší autonomně na základě zadaného grafu přechodu, návrhář tedy nemusí další syntézu provádět.

Graf přechodu automatu typu Mealy je na Obr. 62. Uzly grafu odpovídají stavům (Q), hrany grafu možným přechodům mezi stavy (δ) a ohodnocení hran odpovídá vstupním podmínkám nutným pro přechod mezi stavy (X) a odpovídajícímu výstupu (Y). Odpovídající tabulka přechodů a výstupů je v Tab. 20.



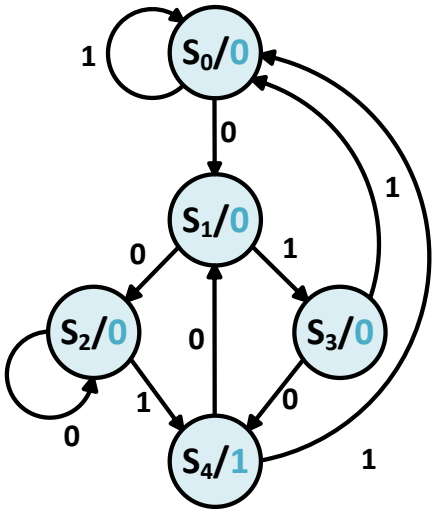
Obr. 58: Graf přechodů Mealyho automatu

Tab. 32: Tabulka přechodů a výstupů Mealyho automatu

Stav Q_t	Vstup N	Budoucí stav Q_{t+1}	Výstup Z
S0	0	S1	0
	1	S0	0
S1	0	S2	0
	1	S3	0
S2	0	S2	0
	1	S0	1
S3	0	S0	0
	1	S0	1

Graf přechodu automatu typu Moore je na Obr. 63. V uzlech grafu jsou vepsány kromě pojmenování odpovídajícího stavu i hodnoty příslušného výstupu (Y). Odpovídající tabulka

přechodů a výstupů je v Tab. 21.



Obr. 59: Graf přechodů Mooreova automatu

Tab. 33: Tabulka přechodů a výstupů Mooreova automatu

Stav Q_t	Vstup N	Budoucí stav Q_{t+1}	Výstup Z
S0	0	S1	0
	1	S0	0
S1	0	S2	0
	1	S3	0
S2	0	S2	0
	1	S4	0
S3	0	S0	0
	1	S4	0
S4	0	S1	1
	1	S0	1

Při zadávání sekvenčního systému se často stane, že tabulka přechodů a výstupů obsahuje vzájemně ekvivalentní stavy. Tyto stavy mají pro všechny vstupní symboly definovány přechody do stejného stavu (nebo do stavů o kterých víme, že jsou ekvivalentní) a zároveň všechny výstupní symboly jsou pro všechny přechody stejné. Ekvivalentní stavy můžeme nahradit jediným representantem celé třídy ekvivalentních stavů. Dojde tak ke snížení počtu stavů automatu a tím často ke snížení počtu klopných obvodů automatu při zachování stejné funkčnosti.

8.3 Převody mezi automaty

Automaty Mealy a Moore jsou navzájem na sebe převoditelné, jestliže mají shodnou vstupní i výstupní abecedu. Znamená to, že každý z těchto automatů je možné nahradit typem druhým, přičemž pro libovolnou vstupní sekvenci dostaneme shodnou sekvenci výstupní jako u automatu původního.

Přeměna automatu Moore na automat Mealy:

Mějme automat A typu Moore charakterizovaný pětící <X, Y, Q, δ, λ>. Jestliže chceme vytvořit ekvivalentní automat typu Mealy můžeme využít stejnou množinu stavů Q a stejnou přechodovou funkci mezi stavy δ. Podle definice chování automatu vidíme, že výstupní funkce λ se musí lišit,

protože má jiné vstupní parametry. Označme tedy výstupní funkci automatu Mealy symbolem λ´. Automat Mealy s označením A´ bude tedy charakterizován pěticí <X, Y, Q, δ, λ´>. Aby oba automaty byly ekvivalentní (A=A´) musí platit:

$$\lambda´((X(t), Q(t-1))= \lambda (\delta (X(t), Q(t-1))) = \lambda Q(t) = Y(t)$$

V Tab. 22 a 23. jsou tabulky přechodů a výstupů obou ekvivalentních automatů. Vidíme, že přechody mezi stavy jsou shodné a výstupy automatu Mealy pro jednotlivé stavy a vstupy odpovídají výstupům automatu Moore pro stavy, do kterých by automat přešel v následujícím taktu při aplikování příslušného vstupu.

Tab. 34: Tabulka přechodů a výstupů Mooreova automatu (ekvivalentní s Mealym)

Přechody automatu			Výstup
X/Q	X1	X2	Z
Q1	Q3	Q1	Y3
Q2	Q1	Q2	Y1
Q3	Q2	Q3	Y2

Tab. 35: Tabulka přechodů a výstupů Mealyho automatu (ekvivalentní s Moorem)

Přechody automatu		Hodnota výstupu při vstupech X1 a X2	
Q/X	X1 X2	X1	X2
Q1	Q3 Q1	Y2	Y3
Q2	Q1 Q2	Y3	Y1
Q3	Q2 Q3	Y1	Y2

V případě, že přeměňujeme Mooreův automat na Mealyho může dojít k tomu, že bude obsahovat ekvivalentní stavy, takže nebude minimální. V tom případě je možné v dalším kroku provést redukci stavů.

8.3.1 Přeměna automatu Mealy na Moore

Automat typu Mealy může mít nedostatečný počet vnitřních stavů k tomu, aby bylo možné splnit podmínku ekvivalence. Tato podmínka je splnitelná tehdy, když pro všechny stavy platí, že do každého uzlu grafu přechodu vedou pouze hrany ohodnocené stejným výstupním symbolem. Tato podmínka není splněna např. v příkladu na Obr. 62, kde do uzlu A vstupují hrany ohodnocené různě. Při vytváření ekvivalentního grafu automatu Moore nahradíme každý uzel, který nemá uvedenou vlastnost tolika uzly, kolika výstupními symboly jsou ohodnoceny hrany do něho vstupující. Uzly ohodnotíme příslušnými výstupními symboly, které odpovídají výstupním symbolům uvedeným u příslušných vstupujících hran. Nakonec připojíme vstupní a výstupní hrany těch uzlů, které jsme doplnili, tak, aby byla zachována návaznost vnitřních stavů automatu. Příklad automatu Moore, který je ekvivalentní Mealyho automatu z Obr. 62 je na Obr. 63.

8.4 Postup syntézy automatu

Postup návrhu automatu sestává z následujících kroků:

- Volba typu automatu
- Sestavení grafu přechodů automatu

- Sestavení tabulky přechodů a výstupů
- Redukce stavů
- Přiřazení stavů automatu vnitřním proměnným navrhovaného obvodu
- Sestavení budící tabulky pro zvolené typy klopných obvodů
- Návrh propojení obvodu

Volba typu automatu

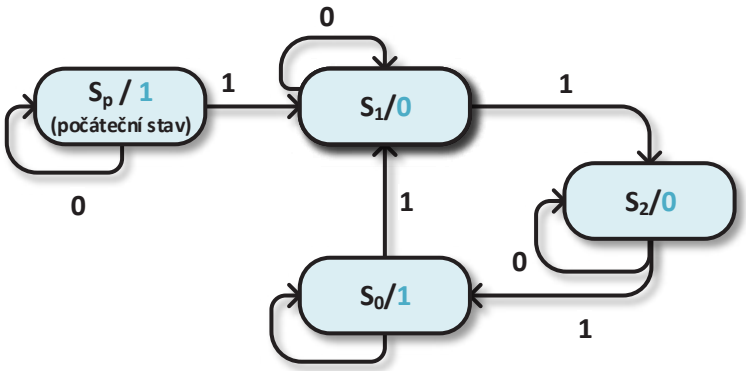
Automat typu Mealy je konstrukčně jednodušší, Mooreův automat je lépe synchronizován. Volba typu tak závisí na podmínkách využití automatu, na součástkové základně, která bude použita pro konstrukci automatu.

Sestavení grafu přechodů automatu

Graf přechodu automatu sestavíme podle slovního zadání, které má charakterizovat požadované výstupní sekvence pro dané sekvence vstupů. Postup budeme demonstrovat na příkladu.

Slovní zadání: Sestavte čítač modulo 3. Čítač v případě vstupního signálu N rovném log. 1 cyklicky generuje na svém výstupu Y log. 1 v případě, že počet hodinových taktů od počátku činnosti je roven 0 modulo 3. V ostatních případech je na výstupu log. 0. Graf přechodů je vyobrazen na Obr. 64.

Na grafu přechodu vidíme 4 stavy, z nichž jeden je počáteční a dalšími třemi bude automat cyklicky procházet, při průchodu stavem S0 bude na výstupu generovat log. 1.



Obr. 60: Graf přechodů automatu pracujícího jako čítač modulo 3

Sestavení tabulky přechodů a výstupů

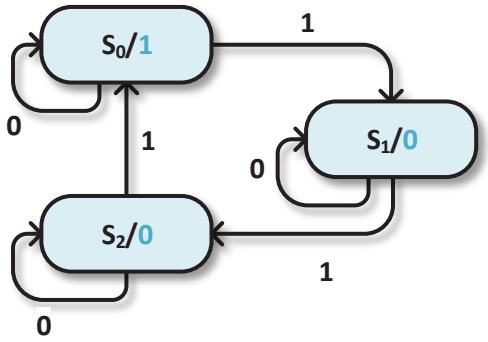
Tabulka odpovídající grafu přechodu automatu je v Tab. 25.

Tab. 36: Tabulka přechodů a výstupů automatu pracujícího jako čítač modulo 3

stav Q_t	vstup N	budoucí stav Q_{t+1}	výstup Y
Sp(počáteční stav)	0	Sp	1
	1	S1	
S0	0	S0	1
	1	S1	
S1	0	S1	0
	1	S2	
S2	0	S2	0
	1	S0	

Redukce stavů

Tab. 24 obsahuje dva ekvivalentní stavy. Jedná se o stav počáteční a stav S0. Důvodem pro toto tvrzení je fakt, že oba stavy mají následující stavy stejné a zároveň mají přiřazené shodné výstupy. Z toho důvodu odstraníme stav nazvaný počáteční. Výsledný graf přechodu a tabulka přechodu a výstupů je na Obr. 65 resp. Tab. 25.



Obr. 61: Redukovaný graf přechodů automatu pracujícího jako čítač modulo 3

Tab. 37: Redukovaná tabulka přechodů a výstupů automatu pracujícího jako čítač modulo 3

stav Q_t	vstup N	budoucí stav Q_{t+1}	výstup Y
S0	0	S0	1
	1	S1	
S1	0	S1	0
	1	S2	
S2	0	S2	0
	1	S0	

Přiřazení stavů automatu vnitřním proměnným navrhovaného obvodu

Jednotlivé stavy automatu jsou representovány stavy klopných obvodů. Obecně platí, že je třeba použít tolik klopných obvodů, kolik jich je třeba k zakódování stavů vhodným kódem. Používá se binární kód, Grayův kód, kód 1 z n a další. Vhodnou volbou kódu je možné redukovat hardwarové nároky na realizace kombinačních funkcí λ a δ.

Ve sledovaném příkladu je možné využít 2 klopných obvodů a jako kód zvolit binární kód. Výsledné přiřazení hodnot klopných obvodů stavům je v Tab. 27. Tabulka přechodů a výstupů obsahující informaci o zakódovaných vnitřních stavech klopných obvodů je v Tab. 28.

Tab. 38: Přiřazení hodnot klopných obvodů vnitřním stavům automatu

stav Qt	Vnitřní proměnné	
	A	B
S0	0	0
S1	0	1
S2	1	0

Tab. 39: Tabulka přechodů a výstupů automatu s přiřazenými stavy

Q_t	A	B	N	Q_{t+1}	A	B	Y
S0	0	0	0	S0	0	0	1
			1	S1	0	1	

Q_t	A	B	N	Q_{t+1}	A	B	Y
S1	0	1	0	S1	0	1	0
			1	S2	1	0	
S2	1	0	0	S2	1	0	0
			1	S0	0	0	

Sestavení budící tabulky pro zvolené typy klopných obvodů

Podle Obr. 61 výstup kombinační funkce δ logicky napájí vstupy klopných obvodů. Požadovaný stav klopných obvodů z Tab. 28 docílíme pouze tehdy, když na vstupy přivedeme takové hodnoty, které zaručí v následujícím taktu tento stav. V Tab. 29 jsou uvedeny pro všechna možná překlopení různých klopných obvodů hodnoty vstupů, které je třeba nastavit.

Tab. 40: Budící tabulka klopných obvodů

Současný stav	Následující stav	D klopný obvod	JK klopný obvod		T klopný obvod
Q_t	Q_{t+1}	D	J	K	T
0	0	0	0	X	0
0	1	1	1	X	1
1	0	0	X	1	1
1	1	1	X	0	0

Volba typu klopného obvodu je určena často součástkami, které má návrhář k dispozici. Obecně je možné říci, že realizace s JK klopnými obvody bývá náročnější na počet kombinačních funkcí, které je třeba vypočítat. Realizace s D klopnými obvody je výhodná spíše pro automaty, které klasifikují nějakou vstupní posloupnost, a realizace s T popřípadě JK obvody je vhodná tehdy, když automat čítá nějaké vstupní události.

V našem příkladu ukážeme výsledné logické výrazy funkce δ pro všechny 3 případy realizace automatu: s klopnými obvody typu D, T a JK. Tab. 30, Tab. 31. Tab. 32 doplňují předchozí tabulky o sloupce s budícími vstupy těchto klopných obvodů. Sloupce DA a DB odpovídají realizaci pomocí D klopných obvodů, sloupce JA, KA, JB a KB odpovídají obvodům JK a sloupce TA a TB obvodům T.

Tab. 41: Tabulka přechodů a výstupů s hodnotami buzení klopných obvodů typu D

Stav		Vstup	Budoucí stav			Buzení KO		Výstup
Q_t	A	B	N	Q_{t+1}	A	B	DA	DB	Y
S0	0	0	0	S0	0	0	0	0	1
			1	S1	0	1	0	1	
S1	0	1	0	S1	0	1	0	1	0
			1	S2	1	0	1	0	
S2	1	0	0	S2	1	0	1	0	0
			1	S0	0	0	0	0	

Tab. 42: Tabulka přechodů a výstupů s hodnotami buzení klopných obvodů typu JK

Q_t	A	B	N	Q_{t+1}	A	B	JA	KA	JB	KB	Y
S0	0	0	0	S0	0	0	0	X	0	X	1
			1	S1	0	1	0	X	1	X	
S1	0	1	0	S1	0	1	0	X	X	0	0

Q_t	A	B	N	Q_{t+1}	A	B	JA	KA	JB	KB	Y
			1	S2	1	0	1	X	X	1	
S2	1	0	0	S2	1	0	X	0	0	X	0
			1	S0	0	0	X	1	0	X	

Tab. 43: Tabulka přechodů a výstupů s hodnotami buzení klopných obvodů typu T

Q_t	A	B	N	Q_{t+1}	A	B	TA	TB	Y
S0	0	0	0	S0	0	0	0	0	1
			1	S1	0	1	0	1	
S1	0	1	0	S1	0	1	0	0	0
			1	S2	1	0	1	1	
S2	1	0	0	S2	1	0	0	0	0
			1	S0	0	0	1	0	

Pro jednotlivé zvolené typy klopných obvodů tyto sloupce odpovídají funkčním hodnotám příslušných kombinačních obvodů funkce δ přičemž odpovídajícími vstupními hodnotami jsou stavy klopných obvodů a hodnota vstupu v příslušném řádku. Popisy těchto kombinačních funkcí můžeme vyextrahovat do Karnaughových map nebo můžeme rovnou sestavit minimální logické výrazy, které representují. Příslušné logické výrazy jsou:

$$D_A = \overline{A}\cdot B\cdot N + A\cdot\overline{B}\cdot\overline{N}$$

$$D_B = \overline{A}\cdot\overline{B}\cdot N + \overline{A}\cdot B\cdot\overline{N}$$

$$J_A = \overline{A}\cdot B\cdot N$$

$$K_A = N$$

$$J_B = \overline{A}\cdot\overline{B}\cdot N$$

$$K_B = N$$

$$T_A = \overline{A}\cdot B\cdot N + A\cdot\overline{B}\cdot N$$

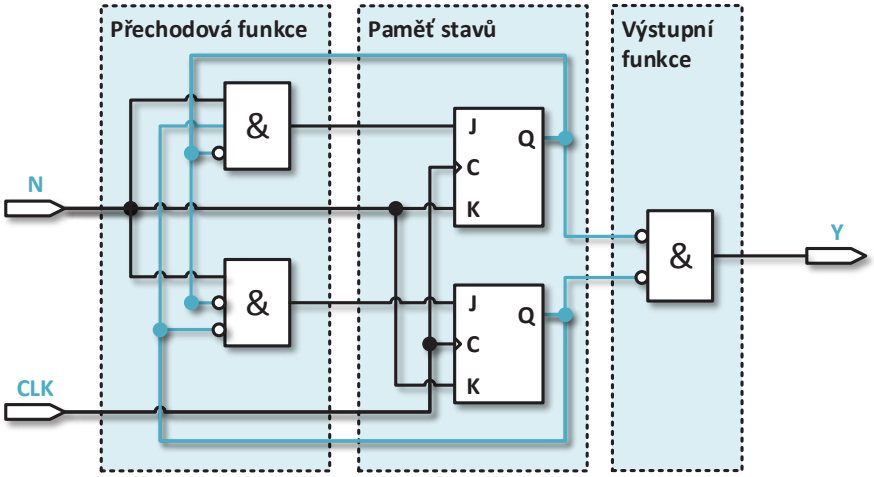
$$T_B = \overline{A}\cdot\overline{B}\cdot N + \overline{A}\cdot B\cdot N$$

$$Y = \overline{A}\cdot\overline{B}$$

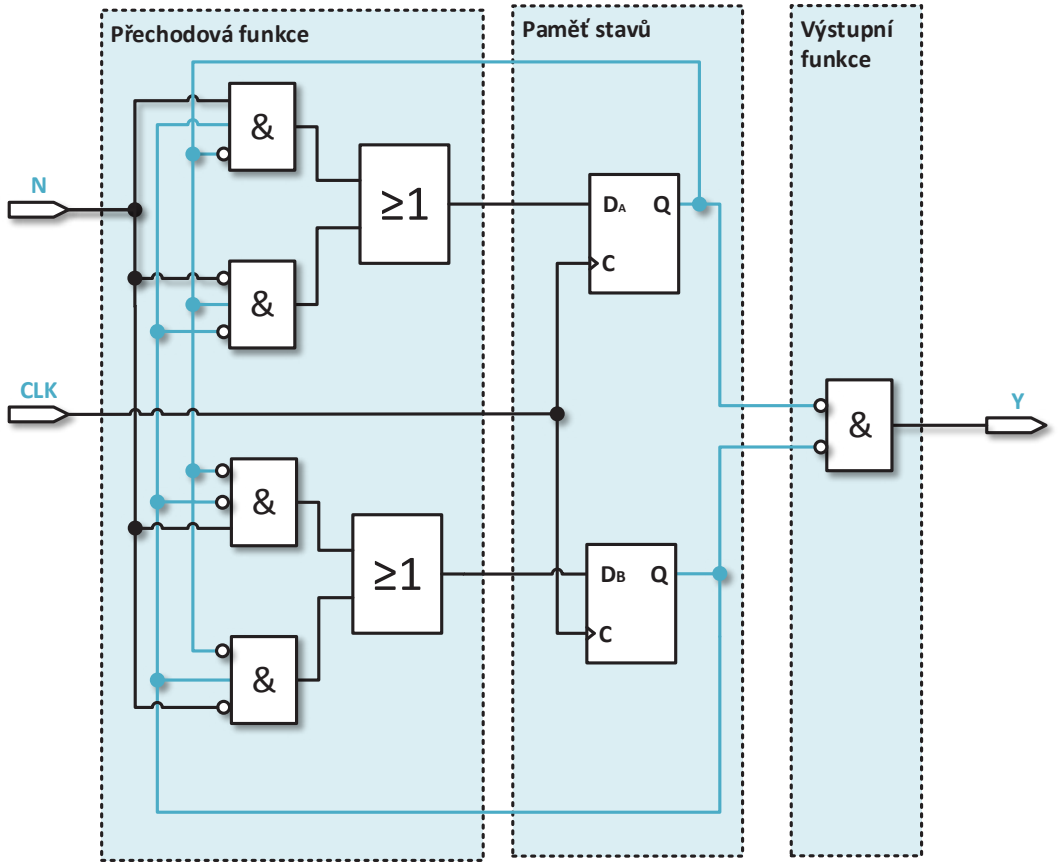
Tyto logické výrazy byly přímo vyčteny z Tab. 30 až 32.

Návrh propojení obvodu

Realizace automatu pomocí obvodů JK, D a T jsou vyobrazeny na Obr. 66, Obr. 67 a Obr. 68.

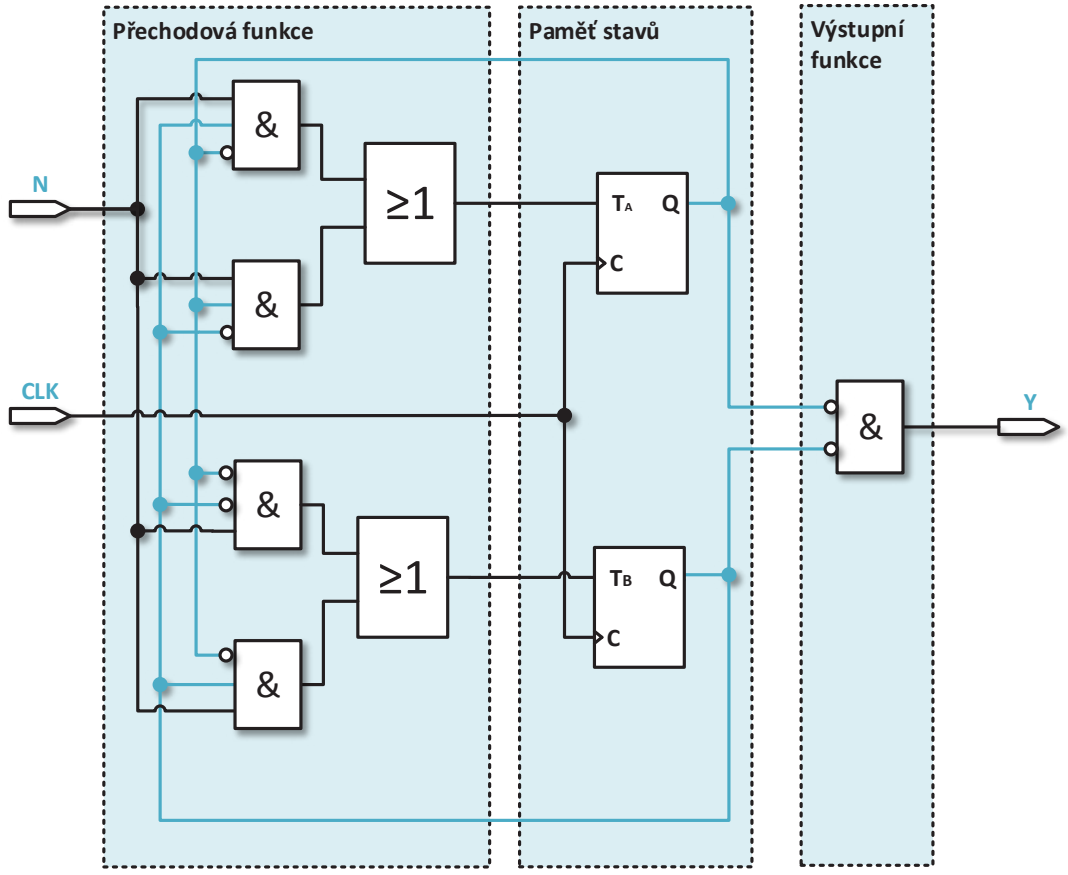


Obr. 62: Schéma zapojení automatu s využitím JK klopných obvodů



Obr. 63: Schéma zapojení automatu s využitím D klopných obvodů

Pro zapojení jsme využili logických výrazů získaných z tabulek.



Obr. 64: Schéma zapojení automatu s využitím T klopných obvodů

8.4.1 Možné problémy realizace automatů

V ilustračním příkladu jsme minimalizovali počet stavů na 3. Vzhledem k tomu, že tento počet stavů lze zakódovat 2 klopnými obvody zůstává otázka, co způsobí 4. možný stav, který není uvažován pro funkci automatu. Do tohoto stavu může automat přejít například v okamžiku zapnutí napájecího napětí. Jestliže není přechodovou funkcí δ definován deterministický přechod z tohoto stavu do nějakého vnitřního stavu automatu, mohlo by se stát, že se automat zablokuje anebo bude generovat neplánovanou sekvenci výstupů. To lze řešit dvěma způsoby:

- vybavit automat asynchronním resetem a zajistit tak vždy při jeho aktivaci uvedení do předem definovaného stavu
- doplnit graf přechodu automatu o všechny další možné stavy a počítat tak s možností definovaného přechodu automatu z každého nevyužitého stavu.

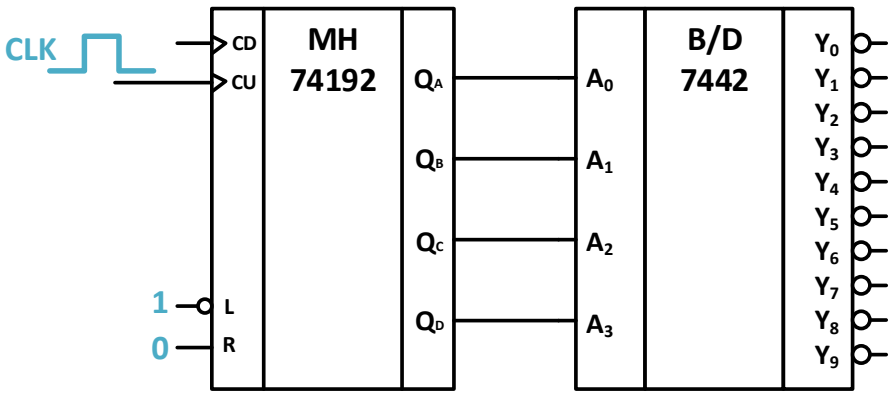
Druhou možnost bychom v ilustračním příkladu využili tehdy, kdybychom neprováděli vyloučení ekvivalentního stavu „počáteční“ a definovali přechody z něj např. podle grafu na Obr. 64.

V dalším textu využijeme pro syntézu obvodu čítače s možností zkrácení cyklu popř. s možností rozšíření rozsahu, ke kterému doplníme kombinační obvody tak, aby bylo realizováno požadované chování.

Výstupem čítače je stav vyjádřený hodnotou výstupních vodičů. Tento stav můžeme využít jako adresu, jejíž obsah se cyklicky mění, každý následující hodinový impuls zvýší (sníží) dekadickou hodnotu adresy o jednotku popřípadě nastaví počáteční hodnotu cyklu. Je-li hodinový signál získáván z generátoru, obsah adresy se mění pravidelně, taktování čítače je však také možno provádět buď manuálně, nebo pomocí dvojkového výstupu nějakého čidla, které měří libovolnou fyzikální veličinu. Adresa může být zpracována v kombinačních obvodech tak, že každému stavu bude odpovídat nějaká kombinace výstupních proměnných. Tímto způsobem vytvoříme automat Mooreova typu [2].

Př. Nn: Rozsvěcujte postupně 8 LED diod tak, aby se po rozsvícení a zhasnutí jedné diody rozsvítila další, atd. Rozsvěcení cyklicky opakujte.

Pro řešení předpokládejme použití čítače 74192, který cyklicky generuje 10 za sebou jdoucích adres. Výstup čítače je vhodné propojit s dekodérem 7442 (binárně desítkový dekodér). Každý následující hodinový impuls přenese log 0 na následující výstup dekodéru. Připojíme-li na výstupy dekodéru LED diody, jejichž druhý vývod připojíme přes rezistory na napětí +5V, pak se po zhasnutí předchozí rozsvítí následující dioda, tj. pohybuje se světelný bod. Rychlost pohybu se nastavuje změnou frekvence hodinového signálu. Jelikož LED diod je pouze 8 a za sebou generujeme vždy 10 adres, dochází k časové prodlevě na konci řady. V tomto případě je vhodné volit např. čítání vzad v neúplném cyklu od 8 do 1 a nepoužít výstupu dekodéru číslo 0.



Obr. 65: Spojení čítače a dekodéru

Sestavte obvod ovládající semafor pro auta a semafor pro chodce na řízeném přechodu pro chodce.

Postup: Nejprve připravíme pravdivostní tabulku kombinačních funkcí. Výstupní proměnné určíme následovně: Č – červená pro auta, Z – zelená pro auta, O – oranžová pro auta, ČCH – červená pro chodce, ZCH – zelená pro chodce. Vstupními proměnnými kombinačních funkcí budou tři výstupy šestnáctkového čítače: a, b c. Pravdivostní tabulka úlohy je v Tab. 30.

Tab. 44: Řízení světelné křižovatky

abc	Č	O	Z	ČCH	ZCH
000	1	0	0	0	1
001	1	0	0	0	1
010	1	0	0	0	1
011	1	1	0	1	0
100	0	0	1	1	0
101	0	0	1	1	0
110	0	0	1	1	0
111	0	1	0	1	0

V pravdivostní tabulce jsme zohlednili požadavek na nestejnou dobu nastavení jednotlivých barev na semaforu. Stavy semaforu, kdy je zelená pro auta a zelená pro chodce se třikrát za sebou v pravdivostní tabulce opakují. Jestliže použijeme k vytvoření signálů a, b, c čítač buzený hodinovým signálem, budou tyto stavy trvat třikrát déle než přechodové stavy, kdy svítí oranžová barva.

Pomocí Karnaughovy mapy a De Morganových vztahů určíme logické výrazy pro jednotlivé výstupní proměnné, při úpravě s výhodou využíváme již určených výstupních proměnných (skupinová minimalizace):

$$Č = a$$

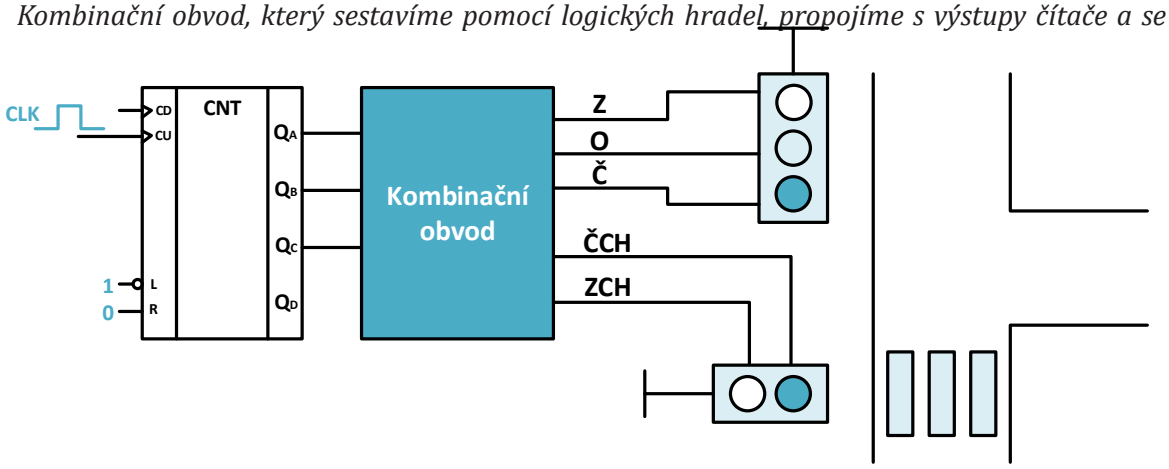
$$O = a \cdot b$$

$$Z = \overline{a} \cdot b + \overline{b} \cdot c$$

$$ČCH = c + ab = O + c$$

$$ZCH = \overline{ČCH}$$

Kombinační obvod, který sestavíme pomocí logických hradel, propojíme s výstupy čítače a se



Obr. 66: Zapojení obvodu řízení semaforů

vstupy semaforů podle Obr. 70. Úlohu můžeme řešit také spojením čítače s dekodérem a přídavnými hradly.

8.4.2 PŘÍKLADY K PROCVIČOVÁNÍ:

Příklad nn: Pomocí čítače určete střední počet impulsů vznikajících při sepnutí a rozepnutí neošetřeného tlačítka a přepínače.

Postup:

1. Neošetřené tlačítko připojíme na hodinový vstup prvního stupně dvoustupňového desítkového čítače v režimu čítáni vpřed. Výstup čítače připojíme na dvoumístný sedmisegmentový displej. Vstup R přivedeme též na tlačítko a tím zajistíme možnost nulování výstupu.

2. Ke každému stlačeni tlačítka (přepnutí přepínače) odečteme výstup čítače a pak jej vynulujeme. Určíme střední hodnotu a směrodatnou odchylku z 10 až 20 pokusů.

Příklad nn: Sestavte sekvenční obvod, který zajistí předepsané cyklické zhasínání a rozsvěcení světelné diody. Obvod využívá multiplexoru.

Postup: Zapojíme čítač s multiplexorem tak, aby se umožnilo cyklické nastavování všech adres multiplexoru. Obsah informačních vstupů multiplexoru nastavujeme na přepínačích tak, že lze cyklus rozsvěcení diod měnit za provozu.

Navrhněte synchronní sekvenční obvod se vstupem x a s výstupem z.

Obvod porovnává bity vstupujícího slova. Výstup z = 1 pouze tehdy, když se na vstupu objeví posloupnost 111. Výstup z se navrátí do nuly zároveň s první 0 na vstupu x.

Navrhněte synchronní sekvenční obvod s jedním vstupem x a s jedním výstupem z.

Obvod po přivedení hodnoty log. 1 na vstup x vygeneruje na výstupu z posloupnost 1010 nezávisle na dalších hodnotách přivedených na x.

Navrhněte synchronní sekvenční obvod s 5 vstupy x1, x2, x3, x4 a x5 a s jedním výstupem z.

Obvod porovnává vstupující 5 bitová slova. V počátečním stavu je z = 0. Jestliže se objeví na vstupech kombinace 10101 automat začne generovat na výstupu z nekonečnou sekvenci 101010.

Navrhněte synchronní sekvenční automat typu Mealy se vstupem x a s výstupy y, z. Obvod porovnává bity vstupujícího slova.

Výstup z = 1 pouze tehdy, když se na vstupu objeví posloupnost 111 , výstup y =1 pro vstupní posloupnost 110.

Navrhněte synchronní sekvenční obvod se vstupem x a s výstupem z. Obvod sčítá jedničky ve vstupující posloupnosti.

Od okamžiku, kdy je počet jedniček větší nebo roven 4 nastaví obvod výstup z = 1.

Navrhněte synchronní sekvenční obvod se dvěma vstupy x1, x2 a jedním výstupem z. Obvod porovnává dvě sériově vstupující slova.

Výstup z = 1 pouze tehdy, když se na obou vstupech zároveň objeví shodné hodnoty (x1 = x2) po dobu alespoň dva hodinové takty. Výstup z se navrátí do nuly zároveň s první rozdílnou dvojicí bitů na vstupech.

Navrhněte synchronní sekvenční obvod se třemi vstupy x1, x2, x3 a s jedním výstupem z.

Obvod porovnává sériově vstupující 3 bitová slova.

Výstup z = 1 pouze tehdy, když se na všech vstupech zároveň objeví shodné bity (x1=x2=x3) po dobu alespoň 2 hodinových pulsů . Výstup z se navrátí do nuly v okamžiku, kdy podmínka shodných bitů přestane platit.

Navrhněte synchronní sekvenční obvod se vstupem x a s výstupem z.

Obvod sčítá jedničky ve vstupující posloupnosti. Od okamžiku, kdy je počet jedniček větší nebo roven 3 nastaví obvod výstup z = 1.

Navrhněte synchronní sekvenční obvod se vstupem x a s výstupem z.

Obvod po přivedení hodnoty log. 1 na vstup x vygeneruje na výstupu z posloupnost 1010 nezavisle na dalších hodnotách přivedených na x.

Navrhněte synchronní sekvenční automat typu Moore se vstupem x a s výstupy y, z.

Obvod porovnává bity vstupujícího slova. Výstup z = 1 pouze tehdy, když se na vstupu objeví posloupnost 11 , výstup y =1 pro vstupní posloupnost 10.

---

9 Paměti

Za paměť lze označit libovolné zařízení, které umožňuje uložení, uchování a znovunačtení dat. Paměti realizované s využitím křemíku (bez ohledu na technologii) nazýváme polovodičové paměti. Tyto paměti jsou dnes nedílnou součástí většiny elektronických zařízení. Nejjednodušší paměťové prvky jsou klopné obvody (viz kapitola nn), které se používají k uchování jednobitové informace. Vedle procesoru je paměť součástí každého počítače. Informace je do paměti ukládána a z paměti vybírána po malých částech rozsahu několika bitů. Oblast paměti, která obsahuje tuto základní jednotku informace, nazveme buňkou. Každé buňce paměti je přiřazena určitá adresa a zavedením adresy na adresové vstupy paměti můžeme pracovat s obsahem příslušné buňky. Polovodičové paměti se rozdělují podle několika hledisek:

- **Podle závislosti na napájecím napětí**
  - **Volatilní** - po odpojení napájecího napětí ztrácí svůj původní obsah
  - **Nonvolatilní** - jsou nezávislé na napájecím napětí

- **Podle technologie**
  - **Bipolární** – rychlé, vyšší příkon, nižší hustota integrace
  - **Unipolární** – vyšší hustota integrace, v současnosti nejčastěji používané

- **Podle možností zápisu a čtení dat**
  - **Paměti pouze pro čtení**
    - **ROM** (Read Only Memory) – obsah paměti je určen při výrobě, tento obsah není možné měnit
    - **PROM** (Programmable ROM) – obsah paměti může být jednou naprogramován uživatelem
  - **Paměti převážně pro čtení**
    - **EPROM** (Erasable PROM) – elektricky programovatelná paměť s možností opakovaného programování (umožňuje pouze několik set cyklů). Vymazání paměti je prováděno UV záření přes okénko v pouzdru (vyšší cena pouzdra)
    - **EEPROM** (Electrically Erasable PROM) – elektricky programovatelná a smazatelná paměť. Tento typ paměti umožňuje za provozu programovat a mazat některé buňky. Přepisování dat v paměti je prováděno ve speciálním režimu se zvýšeným napětím. Nástupcem pamětí EEPROM jsou rychlé paměti FLASH umožňující velmi rychlý přepis dat v normálním režimu.
  - **Paměti pro zápis a čtení**
    - **RWM** (Read Write Memory) – paměti umožňující libovolné množství čtení a zápisů (v libovolnou dobu stejnou rychlostí) za běžného provozu. Většinou se jedná o volatilní paměti, takže po odpojení napájení ztrácí svůj obsah. Po znovuzapojení obsahují náhodná data.

- **Podle přístupu k paměti (Obr. nn)**
  - **RAM** (Random Access Memory) – paměť s libovolným přístupem
  - **SAM** (Seriál Access Memory) – přístupové adresy jsou generovány sekvenčně (např.

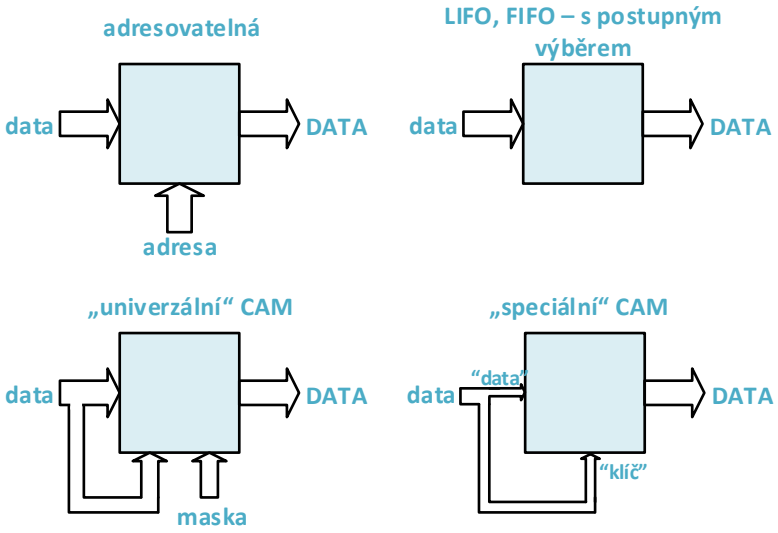
vyrovnávací paměti grafických karet)

- **CAM** (Content Adressable Memory) – výběr adresy podle části uložené informace
- **LIFO** (Last In First Out) – zásobník
- **FIFO** (First In First Out) – fronta

- **Podle způsobu uchování informace**

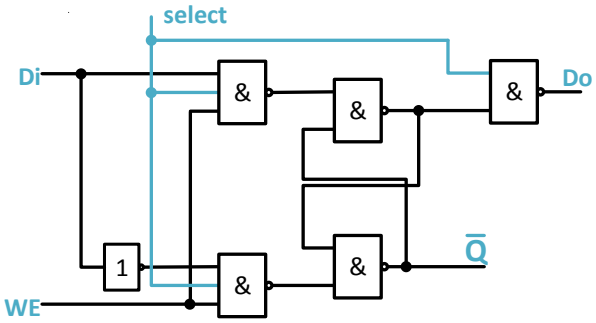
- **SRAM** (Statické RWM – termín RWM se příliš nevžil proto název SRAM respektive DRAM) – paměťové buňky jsou založeny na jednostupňovém D klopném obvodu. Statická paměť má podstatně vyšší spotřebu při srovnatelné frekvenci zápisu dat (oproti dynamické paměti).

- **DRAM** (Dynamické RWM) – u těchto pamětí je informace uchovávána v podobě náboje u řídicí elektrody tranzistoru MOS. Velikost kapacity je v řádu desetin pF (s časem zaniká) informaci v paměti je tedy třeba periodicky obnovovat. Dynamická paměť má delší dobu přístupu k datům, snáze se však integruje (oproti statické paměti)



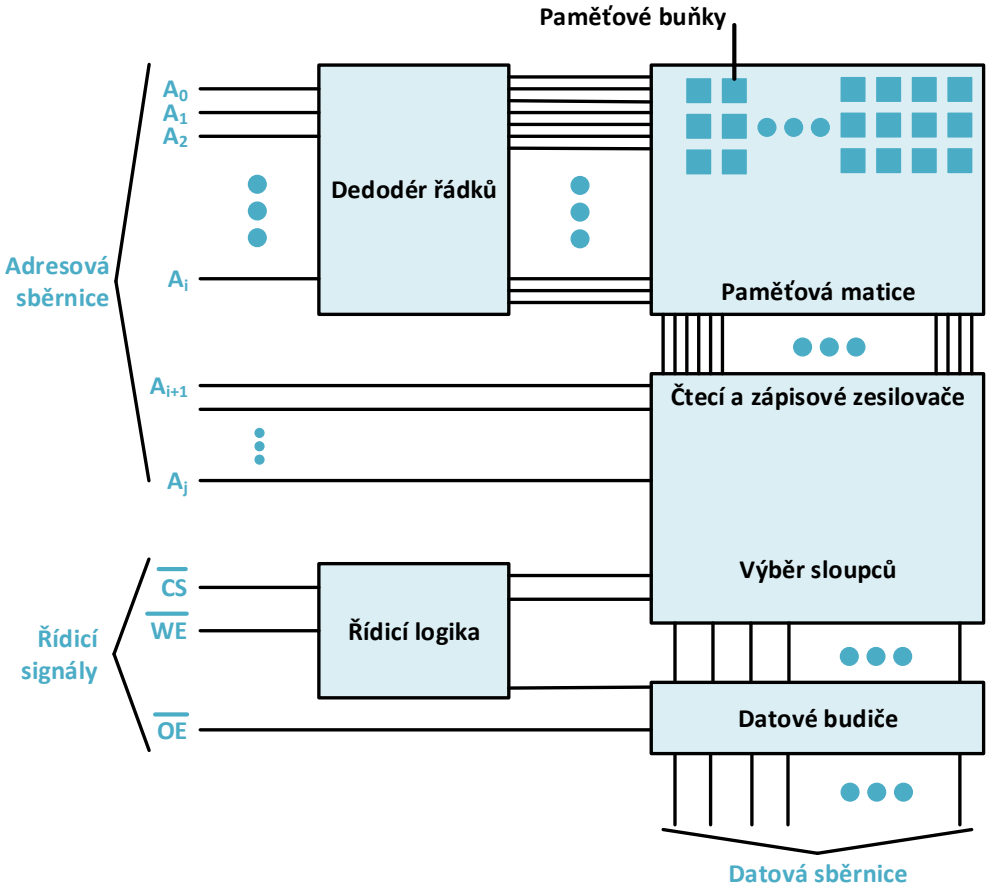
Obr. 67: Blokové naznačení některých přístupů k paměti

Schéma statické paměťové buňky sestavené z hradel NAND můžeme vidět na následujícím obrázku.



Obr. 68: Statická paměťová buňka

Organizaci paměti rozumíme počet bitů, který je k dispozici na dané adrese. Obsah paměti přiřazený adrese nazýváme slovo. Je-li N počet adresovatelných míst a n počet bitů ve slově, pak kapacita paměti je N slov a N.n bitů. Slova bývají jedno, osmi, šestnácti nebo 32-bitová. Ač polovodičové paměti jsou značně různorodé jejich vnitřní strukturu lze obecně popsat (viz následující obrázek).



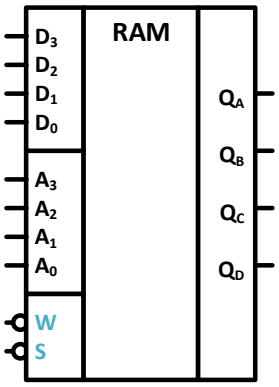
Obr. 69: Schéma uspořádání paměti.

Podstatnou částí paměti je tzv. paměťová matice, která je tvořena paměťovými buňkami. Paměťové buňky jsou uspořádány do sloupců a řádků. V každé paměťové buňce je uložena informace o velikosti 1 bit. Vstupem paměti je adresa, výstupem jsou pak data na této adrese obsažená.

9.1 RAM 7489

V dalším textu si stručně popíšeme paměťový obvod RAM 7489, který je využíván na cvičeních z některých předmětů.

Paměť RAM 7489 polovodičová paměť s kapacitou 64 bitů. Tato paměť umožňuje adresovat čtyřbitová slova na 16 čtyřbitových adresách. Schématická značka paměti je na Obr. mm. Na vstupy A0, A1, A2, A3 se přivádí čtyřbitová adresa, na vstupy D0, D1, D2, D3 čtyřbitová data.



S	W	Akce
0	0	zápis dat
0	1	čtení dat
1	0	přímý přenos dat
1	1	zablokování paměti

Obr. 70: Paměť RAM 7489 a význam signálů S a W

Na výstupech Q0, Q1, Q2, Q3 je obsah adresovaného místa v negovaném tvaru. Význam řídicích vstupů S a W je v pravdivostní tabulce na Obr. mm. Vstup S (select) umožňuje blokování paměti. Tento vstup je aktivní v log.0. Aby bylo možné s pamětí pracovat, je nutné na vstup S přivést log. 0.

Z pravdivostní tabulky na vyplívá, že paměť může pracovat ve 4 režimech:

- **Zápis** - data ustálená na datových vstupech D se zapíši do paměti na adresu nastavenou na adresových vstupech A. Je nastaveno S = 0 , W = 0.
- **Čtení** - obsah paměti na adrese nastavené na adresových vstupech A se v negovaném tvaru přenese na výstup Q.Obsah datových vstupů je ignorován. Je nastaveno S = 0, W = 1.
- **Přenos** - obsah datových vstupů D se po inversi přenese na výstup 4. Obsah adresových vstupů je ignorován. Je nastaveno S = 1,W = 0.
- **Blokování** - na výstupu Q jsou log.1, obsah datových i adresových vstupů je ignorován. Je nastaveno S= 1, W= 1.

Při zápisu dat do paměti se postupuje takto:

a) Na adresových vstupech se nastaví adresa, AO je nejnižší řád adresy.

b) Na datových vstupech se nastaví vkládaná binární informace, DO je nejnižší řád dat.

c) Nastavíme S = 0, W = 0.

d) Na výstupu se objeví zavedená informace v negovaném tvaru.

9.2 Použití pamětí RAM

Paměti RAM nebo ROM lze mj. využit ke generování logických funkci zadaných pravdivostní tabulkou. Nezávislé proměnné tvoří adresu, přiřazené funkční hodnoty v předepsaném pořadí data zapisovaná na tuto adresu. Pokud počet nezávislých logických proměnných není větší než 4 a požaduji se maximálně 4 logické funkce, lze použít paměti RAM 7489. Paměť PROM 74188 umožňuje zpracovat až 8 funkci s 5 logickými proměnnými. Realizace logické funkce paměti RAM sestává ze dvou kroků:

a) zápis pravdivostní tabulky do paměti – programování paměti

b) zobrazeni funkčních hodnot.

Pravdivostní tabulku zapisujeme do paměti RAM řádek po řádku. Na adresový vstup obvodu 7489 přivedeme nezávisle proměnné. Při tom je nutno respektovat, že nejnižší řád je na vstupu A0. Na datovém vstupu nastavíme příslušné funkční hodnoty a režim paměti krátce změníme na zápis. Při zobrazeni funkčních hodnot se na výstup paměti RAM připojují invertory a indikační diody. Na adresovém vstupu se nastaví příslušná kombinace proměnných, paměť udržujeme v režimu čtení a načteme příslušné funkční hodnoty.

Pokud se logická funkce často mění, používá se paměti RAM. To má ten nepříjemný důsledek, že po každém vypnuti zdroje je nutno funkční hodnoty znovu zavést do paměti. Pokud je logický systém odladěn, takže by se logické funkce již neměly měnit, je vhodné je naprogramovat jednou provždy do paměti PROM. Pak je ale již jakákoliv změna logických funkci vyloučena. Přesněji řečeno, lze změnit log.0 na log. 1, obrácený krok není možný. Jinou možností je použít paměť EPROM nebo EEPROM, kterou můžeme opakovaně přeprogramovat.

Spojením čítače a paměti lze realizovat programovatelné periodické řízení. Čítač se přizpůsobuje na adresové vstupy a čítá v úplném nebo neúplném cyklu. Do paměti se na odpovídající adresy naprogramuji předepsané stavy výstupů. Tyto stavy se pak v předepsaném pořadí a periodě cyklicky přenášejí na výstup.

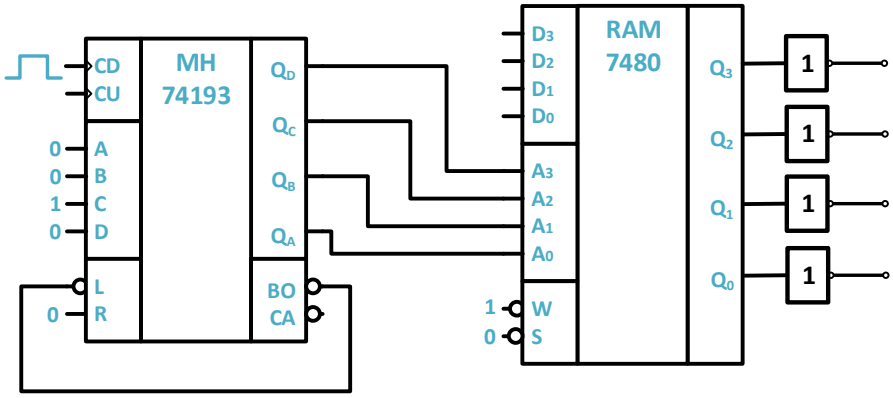
Příklad: Řízení světelné křižovatky

Sestavte obvod ovládající semafor pro auta a semafor pro chodce na řízeném přechodu pro chodce. (0: Řízení světelné křižovatky) K řešení využijte spojení čítače a paměti.

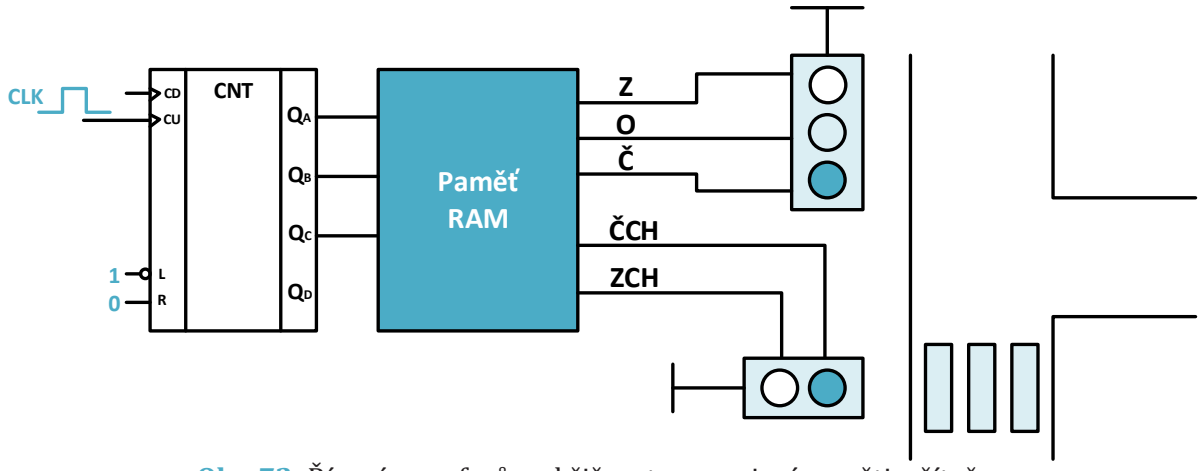
Tab. 45: Pravdivostní tabulka pro řízení světelné křižovatky

abc	Č	O	Z	ČCH	ZCH
000	*1*	*0*	*0*	*0*	*1*
001	*1*	*0*	*0*	*0*	*1*
010	*1*	*0*	*0*	*0*	*1*
011	*1*	*1*	*0*	*1*	*0*
100	*0*	*0*	*1*	*1*	*0*
101	*0*	*0*	*1*	*1*	*0*
110	*0*	*0*	*1*	*1*	*0*
111	*0*	*1*	*0*	*1*	*0*

Postup: Využijeme pravdivostní tabulky z. kk. Proměnná ZCH je inversí proměnné ČCH, a proto stačí nastavovat pouze sekvenci logických hodnot pro proměnné Č, Z, O a ČCH, hodnoty proměnné ZCH získáme inversí signálu ČCH. Z tohoto důvodů můžeme využít paměti 7489, která umožňuje pracovat se čtyřbitovými slovy. Čítač s pamětí propojíme analogicky se zapojením uvedeným na Obr. ll s tím, že není nutné zkracovat cyklus čítače a zároveň stačí využívat pouze tři nižší řády výstupů čítače. V režimu Zápis dat do paměti nahrajeme jednotlivé řádky invertované pravdivostní tabulky. Inversi provádíme z toho důvodu, abychom eliminovali vliv inverse výstupních signálů, která je zabudována ve vlastním paměťovém obvodu. V režimu Čtení dat bude paměť při přivádění adres z čítače postupně generovat sekvenci signálů semaforu. Výstupy obvodu z Obr. ff se propojí k jednotlivým světlům semaforů.



Obr. 71: Spojení paměti a čítače



Obr. 72: Řízení semaforů na křižovatce - spojení paměti a čítače

9.3 3D Paměti

Zvýšení rychlostí a snížení nákladů při výrobě polovodičových pamětí lze například navrstvením několika paměťových čipů na sebe. Díky vrstvení paměťových čipů na sebe v rámci jednoho pouzdra prudce vzrůstá i paměťová kapacita. Přední světoví výrobci paměťových čipů pracují na vývoji a výrobě nového typu přepisovatelných paměťových čipů, označovaných jako 3D paměť. Ta má potenciál stát se budoucím nástupcem v současnosti velmi rozšířených flash pamětí typu NAND. Princip 3D paměti můžeme vidět na následujícím obrázku, kde je vyobrazen čip 3D paměti společnosti IBM.

Obr. jj:

Koncept 3D pamětí není nijak nový. Technologie byla vyvinuta již před několika roky jako způsob, jak snížit cenu a prodloužit dobu uchovávání informací až na 100 let, což je více, než umožňují dnes populární paměti typu NAND.

Vzhledem k neustále se zvyšujícím nárokům na vyšší kapacitu a rychlost a nižší spotřebu a velikost u polovodičových pamětí je skládání paměťových vrstev na sebe (neboli tzv. 3D stacking) jednou z možných budoucích cest vývoje paměťových čipů.

Literatura

[1] Belza, J.: Zapojení s operačními zesilovači. Konstrukční elektronika A-radio, 1 (1996), č.3.

[2] Brandejs, M. Mikroprocesory Intel - Pentium a spol. Grada Praha, 1994 ISBN 80-7169-041-4

[3] Čermák, J. : Kurs polovodičové techniky. SNTL, Praha 1976.

[4] DOLEŽAL, I. a kol. Analogová elektronika. 1. vydání. Liberec: Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, 2014. ISBN 978-80-7494-136-8. DOI: 10.15240/tul/002/2014-11-003

[5] Foit, J. – Hudec, L. : Součástky moderní elektroniky. ČVUT, Praha 1996.

[6] Horowitz, P. – Hill, W.: The Art of Electronics. Cambridge University Press 1989, ISBN 0-521-37095-7

[7] Hrbáček, J. Programování mikrokontrolérů PIC16CXX. Ben Praha, 1998 ISSN 80-86056-16-3

[8] Jáneš, V., Douša, J.: Logické systémy. Skriptum ČVUT, 1995, 299 stran, ISBN 80-01-01106-2

[9] Kubátová, H. Blažek, Z.: Logické systémy cvičení. Skriptum ČVUT, 1996, 105 stran, ISBN 80-01-01227-1

[10] Laipert, M. – Kolář, M. – Horčík, Z.: Systémový návrh zakázkových integrovaných obvodů. Skriptum ČVUT, Praha 1992

[11] Neumann, P. – Uhlíř, J. : Elektronické obvody. [Skriptum] ČVUT, Praha 1996.

[12] Nouza, J. - Doležal, I.. - Košek, M.: Mikroelektronika cvičení. Skriptum VŠST Liberec, 1991, 230 stran, ISBN 80-7083-071-9

[13] Sobotka, Z.: Otázky a odpovede z mikroprocesorov a mikropočítačov. Alfa, Bratislava 1986

[14] Storey, N.: Electronics a systém approach. AW Publishing Comp., 1992, 655 stran, ISBN 0-201-17558-4

[15] Strnad, L.: Základy číslicové techniky, cvičení. Skriptum ČVUT,1999, 124 stran, ISBN 80-01-01433

[16] Vobecký, J. – Záhlava, V. : Elektronika – součástky a obvody, principy a příklady. Grada, Praha 2000.

[17] Zelenka, J. : Elektrotechnika a průmyslová elektronika. [Skriptum] VŠST, Liberec 1983.

[18] Zelenka, J.: Mikroelektronika a měřící technika pro řídící systémy. Skriptum VŠST Liberec, 1991,154 stran, ISBN 80-7083-065-

Název	Číslicová elektronika
Kolektiv autorů	prof. Ing. Ondřej Novák, CSc., Ing. Tomáš Drahoňovský, Ing. Jiří Jeníček, Ph.D., Ing. Zbyněk Mader, Ph.D., Ing. Petr Pfeifer, prof. Ing. Zdeněk Plíva, Ph.D., Ing. Martin Rozkovec, Ph.D.
Určeno	pro studenty TUL
Vydavatel	Technical University in Liberec
Schváleno	Rektorátem TUL dne 23.10.2014, čj.RE 107/14.
Vyšlo	v prosinci 2014
Počet stran	84
Vydání	1. vydání
Číslo publikace	55-107-14
DOI	10.15240/tul/002/2014-11-004

Tato publikace neprošla redakční ani jazykovou úpravou.

ISBN 978-80-7494-137-5